Performance Analysis of Electrical Machines Using a Hybrid Data- and Physics-Driven Model

Vivek Parekh, Dominik Flore, and Sebastian Schöps

Abstract—In the design phase of an electrical machine, finite element (FE) simulations are commonly used to numerically optimize the performance. The output of the FE simulation is used to characterize the electromagnetic behavior of the machine. The simulation workflow involves intermediate measures such as nonlinear iron losses, electromagnetic torque, and flux values at each operating point that are used to compute several key performance indicators (KPIs). We present a data-driven deep learning approach that replaces the computationally heavy FE calculations by a deep neural network (DNN). The DNN is trained by a large volume of stored FE data in a supervised manner. During the learning process, the network response (intermediate measures) is fed as input to a physics-based post-processing to estimate characteristic maps and KPIs. We show that this hybrid approach reduces the computational time while maintaining a high flexibility in the simulation workflow. Finally, the hybrid approach is compared to the existing direct prediction approach for KPIs.

Index Terms—Deep neural network, electrical machine, finite element simulation, key performance indicators.

I. INTRODUCTION

Electrical machines have numerous applications, ranging from home appliances to the automotive industry. Permanent magnet synchronous machines (PMSM) gained popularity in recent years due to their high efficiency, large power density, and high torque-current ratio, see for example Fig. 1. However, the use of expensive materials such as rare earth magnets (neodymium, dysprosium, or terbium) is a significant cost driver in the manufacturing process. Therefore simulation based optimization, e.g. minimizing cost and maximizing performance, is an established technique in academia and industry [1]. The finite element method (FEM) is at the heart of such numerical optimization workflows. In the automotive industry, FEM-based optimization has been used for decades to reduce the number of physical prototypes and to optimize the active parts of the electrical machine (stator, rotor, magnets, winding).

However, it is a computationally expensive process, so there is a great need for complexity reduction.

To this end, surrogate or meta modelling has become a research field on its own in the last decade [2]. More recently, meta-modelling by machine learning has gained attention. Nowadays, deep neural networks (DNNs) are employed at various design stages because of their advantages such as scalability for high dimensional data, handling of Big Data, easy parallelization by graphical processing units (GPUs) and automatic feature extraction. For example, Yan et al. [3] describes the deep-learning-based approximation of the electromagnetic torque for PMSM. In another article, a DNN is used as a meta-model for predicting objectives obtained by FE output in the design phase for the optimization of a flux switching machine to lower the computational burden [4]. Another work [5] demonstrates how a deep learning-assisted approach is used to predict efficiency maps of electric drives. A DNN-based two-step optimization method is investigated in [6] to speed up the design process by inferring torque-related performance. The application of CNN to approximate performance characteristics of electric machines for multidimensional outputs is investigated in [7]. The study in [8] shows how deep learning based meta-models predict cross-domain key performance indicators (KPIs) using different PMSM representations in a large input design space. The meta-model trained with parameter-based
inputs has a higher prediction accuracy than an image-based representation but requires a consistent parameter set. This drawback was mitigated in [9] which demonstrates how KPIs of several machine topologies with incompatible parameterizations can be predicted concurrently by mapping the high dimensional scalar parameters into a lower-dimensional latent space using a variational autoencoder.

In this paper, we present a novel hybrid data-and physics-driven approach for analyzing the performance of an electrical machine. A DNN is used to learn the electromagnetic behavior of an electrical machine. The proposed multi-branch network serves as a meta-model. It accepts a high dimensional input consisting of the relevant machine parameters, e.g., geometry, electrical, and material. The results of the FE simulations are stored in a database, and the DNN is trained by supervised learning. In contrast to previous approaches, in particular [8], [9], we do not directly predict KPIs by a DNN but the intermediate measures. The learning outcome, along with relevant system parameters, is fed into physics-based analytical models to calculate characteristic maps and cross-domain KPIs, e.g., maximum torque, power, cost, etc. Finally, we compare this hybrid model to existing approaches.

The article is organized as follows: Section II explains the problem formulation and data set generation. Section III presents the methodology, network architecture, and training settings. Section IV provides quantitative analysis of results which is followed by the conclusion in Section V.

II. PROBLEM FORMULATION AND DATA SET GENERATION

Each FE simulation requires a high-dimensional parameter-based input \( x \) to compute the intermediate measures \( y \) which then leads to the KPIs \( z \). Let us consider an electric machine simulation dataset

\[
D := \left\{ \left( x^{(1)}, y^{(1)}, z^{(1)} \right), \ldots, \left( x^{(N_D)}, y^{(N_D)}, z^{(N_D)} \right) \right\}
\]

(1)
of \( N_D \) samples, where each input is generated by an input vector \( x^{(i)} \) that consists of random design parameters \( p_j^{(i)} \), constants \( c_k \) and system parameters \( s_l \), where \( j = 1, \ldots, N_p, k = 1, \ldots, N_c \) and \( l = 1, \ldots, N_s \). Examples of those parameters are given in Table I. The FEM model \( F \) computes the \( m \)-dimensional vector of intermediate measures \( y^{(i)} = F(p^{(i)}, c) \). Finally, various physics-based post-processing formulas give rise to the KPIs \( z^{(i)} = K(p^{(i)}, y^{(i)}, s) \). Note that the system parameters \( s \) enter only in the post-processing step. The goal is now to learn a function \( F \) that is cheap to evaluate and approximates the computationally heavy FE simulation. It will be used afterwards as a meta-model to predict intermediate measures for unseen machine designs.

A. Dataset Generation

In this study, the simulation dataset is generated for a double-V PMSM. The procedure is visualized in Fig. 2. In total \( N_p = 35 \) input design parameters \( p_i \) are considered with given upper and lower bounds, e.g., geometry, electrical, and material. A few examples are listed in Table I. Based on this design parameter space an initial population is created using Latin hypercube sampling (LHS) [11]. From this population all geometrically infeasible designs are removed. This leads to a total number of \( N_{EM} = 44877 \) machine designs. Fig. 1 depicts a representative design geometry from this population. Each geometry is then simulated using 2D magneto-static FEM simulations of \( N_{OP} = 37 \) operating points, see e.g. [12]. The operating points are interpreted as a variable electrical excitation input for the electrical machine, i.e., it is given by an input phase current \( I \) and its control angle \( \alpha \). The FEM simulations were run on a high performance computing (HPC) cluster. The CPU time of one simulation of the \( N_{EM} \) designs is about 3-5 hours on a single core machine. Finally, as a result, \( N_D = N_{OP} \times N_{EM} = 1606449 \) samples are available in the dataset \( D \).

The simulated quantities, i.e., intermediate measures \( y^{(i)} \), from the FE analysis are the non-linear iron losses \( P_{fe} \), torque \( T \) and fluxes \( \Psi_1, \Psi_2, \Psi_3 \) linked with the three coils for one electrical period, see Table II. Note that the iron losses \( P_{fe} \) are calculated within the FE code using Steinmetz’s model [13], [14]. The
intermediate measures along with the parameters $p_i$ are post-processed using various physical models to derive the KPIs, see e.g. [15]. A plot of two operating point calculations for one electrical period of a test sample from the dataset $D$ is shown in Fig. 3.

The intermediate measures and other system parameters e.g., the input current $I$, the ohmic resistance $R$ (calculated through winding topology), the number of phases $m$, rotational speed $n_{rpm}$ and geometry parameters like the machine length $l$ are inputs to the post-processing step. For example, the iron-losses can be computed for other desired working points by scaling with respect to rotational speed, material parameters and length. Also the copper loss and shaft power are determined analytically based on physical models

$$P_{CU} = mI^2R \quad \text{and} \quad P_{shaft} = \frac{2\pi n_{rpm}T}{60}.$$  \hspace{1cm} (2)

These KPIs are then used to calculate further KPIs like the efficiency $\eta$ at a certain operation point. Examples of a few KPIs are given in Table III and their distribution along with a few design parameters can be seen in Fig. 4.

### TABLE II

| Measure         | Unit |
|-----------------|------|
| $y_1$ Nonlinear iron losses $P_{IR}$ | J    |
| $y_2$ Electromagnetic torque $T$      | Nm   |
| $y_3$ Flux linkage $\Psi_1$ col 1    | Wb   |
| $y_4$ Flux linkage $\Psi_2$ col 2    | Wb   |
| $y_5$ Flux linkage $\Psi_3$ col 3    | Wb   |

### TABLE III

| KPIs                  | Unit |
|-----------------------|------|
| $x_1$ Maximum torque on limit curve | Nm   |
| $x_2$ Max. shaft power  | W    |
| $x_3$ Max. shaft power at max. speed | W |
| $x_4$ Max. torque ripple on limit curve | Nm |
| $x_5$ Material cost    | Euro |
| $x_6$ Mass of active parts | Kg   |
| $x_7$ Torque ripple deviation | Nm   |

### III. PROCEDURE, NETWORK ARCHITECTURE AND TRAINING

This section discusses the procedure for calculating KPIs using a hybrid approach, followed by specifics on multi-branch DNN architecture and training.

#### A. Procedure

Fig. 5 shows block diagrams of three different approaches for the calculation of KPIs. The classical approach is the conventional way of computing KPIs and is widely used in industry. In this method, as shown in the diagram, the input design parameters $p$ and relevant constants parameters $c$ are fed to the FE simulation, and then the result $y_i$, along with the inputs $p$ and $s$ are given to a physics-based in-house post-processing tool to estimate final KPIs $z_n$. The true functions for computing FE outcomes and KPIs can be written abstractly as

$$z = K(p, F(p, c), s)$$  \hspace{1cm} (3)

Until now classical data-driven approaches, e.g. [8], proposed to learn the function composition $p \rightarrow z$, i.e.

$$z \approx K_z(p)$$  \hspace{1cm} (4)
where ϕ represents DNN model parameters (weights and biases of the DNN). Here, the approximation $\hat{K}_\phi$ works as a meta model to directly predict the KPIs. This model is trained by optimizing the DNN model parameters ϕ, i.e., minimizing the difference between the ground truth and the prediction. The training loss function using the $\ell_1$-norm can be written as

$$\min_\phi L(\phi) := \sum_i \| z^{(i)} - \hat{K}_\phi(p^{(i)}) \|_1$$  \hspace{1cm} (5)$$

where $z^{(i)}$ are true results for the KPIs from $\mathcal{D}$. On the other hand, in our hybrid approach, we use the meta-model only to approximate the computational expensive function $F : p \rightarrow y$ for the FE calculation. Similar as before this approach can be written as

$$z \approx K(p, \hat{F}_\theta(p), s)$$  \hspace{1cm} (6)$$

where now θ represents the DNN model parameters. The training loss function is now

$$\min_\theta L(\theta) := \sum_i \| y^{(i)} - \hat{F}_\theta(p^{(i)}) \|_1$$  \hspace{1cm} (7)$$

where $y^{(i)}$ are true results for the intermediate measures from $\mathcal{D}$.

B. Network Architecture and Training

The performance of deep learning models depends strongly on the choice of hyperparameters. They are fixed before the training, whereas the DNN model parameters, such as biases and weights, are optimized during training. For better understanding, we divide the hyperparameters into two categories: model hyperparameters and learning hyperparameters. The model hyperparameters are specified as the width and height of different network layers, filter size, stride, type of layer (dense,
convolutional, max-pooling, etc.), and so on. The learning hyperparameters include, for example, learning rate, activation functions, optimizer, loss function, batch size, epochs, etc. We propose the multi-branch DNN as depicted in Fig. 6. Initially, this structure was obtained by trial and error, i.e., evaluating more than ten different configurations. The hyperparameters were then numerically optimized. The parameter space is given in Table IV. The hyperparameter optimization (HPO) was performed using an in-house optimization tool that includes implementation of the Asynchronous Successive Halving Algorithm algorithm [16] for the multi-objective case. This tool has evaluated 800 different configurations within the search space defined in Table IV. The entire HPO process took approximately two days with parallelization across four GPUs. The final hyperparameter configuration is given in the last column of Table IV. As shown in Fig. 6, the input layer comprises of varying input design parameters. There are four common layers (CL: 1590 → 1410 → 810 → 220) to exploit correlation among all the output measures. There are a total of five distinct branches with a different number of branch layers (BL). There are two identical branches for torque $\hat{T}$ and iron loss $\hat{P}_{fe}$ prediction, and three equal size branches for flux $\hat{\Psi}$ prediction. The network layers for iron loss and torque are $1530 \rightarrow 1210 \rightarrow 900 \rightarrow 880 \rightarrow 750 \rightarrow 660 \rightarrow 610 \rightarrow 580 \rightarrow 550 \rightarrow 530$. The branch configuration for other three branches are $322 \rightarrow 278 \rightarrow 240$.

As stated earlier in the section Section II-A, each branch predicts intermediate measures for every operating point. In this study, we predict 15-time steps by the flux and torque branches, so the size of output layers remains $15 \times 1$ for both output measures. These figures, however, can be modified as needed. The output layer for iron losses is $4 \times 1$ in size. The total number of trainable network parameters is noted to be around 16.
The actual number of machine designs $N_{EM} = 44877$ is divided into training, validation, and testing. The training sample consists $N_{\text{train}} = 40390$ ($\sim 90\%$), whereas validation $N_{\text{val}} = 2243$ and test samples $N_{\text{test}} = 2244$ comprises $\sim 5\%$ each. Since each machine design is evaluated for all operating points, the total number of training samples is obtained after multiplication by $N_{OP} = 37$.

The data preprocessing is carried out using the sci-kit learn library [17] to further enhance the network’s generalization capability. The model training for the final configuration was carried out on a NVIDIA Quadro M2000 M GPU using standard back-propagation [18]. Tensorflow2 [19], a deep learning framework, was used to implement the training pipeline. It took around 2 hours to finish the training with early stopping criteria (10 epochs over validation error) for a maximum of 300 epochs. The training and validation curve is illustrated in Fig. 8.

For comparison, Fig. 7 shows the network architecture obtained in [8] for the direct approach. The network learning hyperparameters (e.g., learning rate, early stopping criteria, loss function, optimizer, etc.) are kept identical for both networks during network training for the direct and hybrid approaches.

IV. RESULTS AND ANALYSIS

We first discuss the evaluation of the intermediate measures. Following that, we explain the quantitative study with empirical results of the hybrid and direct DL approach. The characterization of the performance of a PMSM is a non-linear multi-output regression problem, and thus we chose to evaluate the performance of meta-models using several evaluation metrics, i.e., the mean relative error (MRE), mean absolute error (MAE), and the Pearson correlation coefficient (PCC).

A. Discussion of the Intermediate Measures Prediction

The problem at hand requires multi-output regression and there are several algorithms and implementations available. We considered several state-of-the-art approaches using the sci-kit-learn library [17], [21] with their default settings. The library contains an implementation of K-Nearest Neighbour (with KNeighborsRegressor) [22], [23], Support Vector Regression [24], and Gaussian Process Regression [25], which were unable to process the data due to memory constraints. These algorithms require higher computational power (ca. five times higher in our experiments) as compared to our existing resources (NVIDIA Quadro M2000 M GPU) to process such a large volume of high dimensional data. Random forest [26] is an ensemble method, that combines multiple decision trees to train a meta-model. The final model makes predictions by taking the average of the results of the different trees (bootstrap aggregation). The algorithm succeeded to train the meta-model using the default settings. However, the prediction accuracy is lower and training time is almost 10 times higher than for a random untuned multi-branch DNN. So, we chose to improve the prediction performance for intermediate measures by optimizing the baseline multi-branch DNN.

| Measure                      | Multi-branch DNN |
|------------------------------|------------------|
| me relative error (MRE)      | $1.5 \times 10^{-3}$ | 0.98 |
| mean absolute error (MAE)    | $5.9 \times 10^{-4}$ | 0.98 |
| hysteresis loss (rotor)      | $1.6 \times 10^{-4}$ | 0.98 |
| hysteresis loss (stator)     | $4.8 \times 10^{-4}$ | 0.99 |

**TABLE V**

**INTERMEDIATE MEASURES OVER TEST SAMPLES WITH OPTIMIZED MULTI-BRANCH DNN**
configuration, we tested the network’s prediction performance at different training-validation-test split percentages (70/20/10, 75/15/10, and 70/10/20) with complete randomization. It has been observed to be robust, i.e., independent of the training-validation-test split. Table V presents numerical accuracy over test samples for the optimized multi-branch network (Fig. 6). We use the MRE to evaluate iron losses because its range varies largely, while torque and flux with MAE. The reason is that the torque $T$ and the fluxes $\Psi_1, \Psi_2, \Psi_3$ are more sensitive. For the testing, the MAE is evaluated over the mean of the predicted time steps for each operating point. Fig. 9 depicts the prediction curve over each operating point for all the test samples of iron losses. It is observed that the multi-branch DNN predicts intermediate measures with high accuracy close to the ground truth for all the test samples. The evaluation time for new machine designs is $\sim 100$ ms/machine design (for all 37 operating points), which is much lower in comparison to the actual FE simulation (3–5 hours/machine design). Fig. 10 illustrates two operating points for different input electrical excitation conditions for one test machine design. Here, the green box shows the time steps that were predicted by multi-branch DNN, while the blue box shows the time steps that were calculated using magnetic state symmetry. Fig. 10(a) unveils that the multi-branch DNN has poor prediction accuracy for the operating point at zero input phase current and open circuit (no load) condition.

**B. Quantitative Analysis**

Let us compare the proposed hybrid approach Fig. 5 with the parameter-based direct DL approach described in [8]. Fig. 14 shows evaluation with MRE over an increasing training size from 5% to the total training size. The hybrid approach consistently performs better for KPIs $z_1, z_2, z_4, z_5, z_6, z_7$, while the direct approach is slightly more accurate for $z_3$. The direct approach shows always monotonic convergence with respect to increasing training data while this is not the case for the first three KPIs in the hybrid approach. However, both meta-models predict KPIs for unseen machine designs. Fig. 13 illustrates the prediction plot over the test samples for meta-models trained on the full training set. As explained in Fig. 5, the training of multi-branch DNN is independent of the system parameters and solely relies on varying geometry, electrical excitation, and material parameters, whereas in the direct DL approach, model training with output KPIs implicitly involve fixed value for the system parameters. This makes the hybrid approach more flexible than the direct DL approach. The post-processing of FE output only takes very little time ($\sim 3–5$ min.) and is performed using a physics-based in-house post-processing tool.
and hence the hybrid approach opens up new possibilities for further analysis. Demonstrative examples for one test sample are shown in Fig. 11 and Fig. 12. Fig. 11 displays different performance curves, e.g., maximum torque curve, open circuit and short circuit voltage characteristic, and maximum shaft power at different rotor speeds. Fig. 12 shows the efficiency map for the given test design, see [27] for a detailed interpretation. The Fig. 12(a) shows the efficiency map for the FE simulation, Fig. 12(b) is calculated from the intermediate measures predicted with multi-branch DNN, while Fig. 12(c) gives details on the deviation between the two. The difference is close to zero with a maximum near the low torque region ($\sim 20\%$). A possible explanation is that multi-branch DNN does not predict well for the open circuit operation mode at zero input current, which is also reflected in Fig. 10(a) for the same associated sample.

In the open circuit mode, the associated torque range for the electrical machine is low ($10^{-1}$ to $10^{-3}$ Nm). If the multi-branch DNN predicts values in the range of $10^{-2}$ to $10^{-4}$ Nm this results in a relatively large error. However, the relatively poor prediction in this region does not have a significant impact on the calculation of the overall efficiency of the electrical machine.
at other operating points (especially high-efficiency operating point regions as shown in Fig. 12(c)). There are also a few disadvantages of the proposed hybrid approach. The training time of the multi-branch DNN is roughly about ~ 2 hours, which is about 6× higher than the training time (~ 20 minutes) of the DNN defined for direct KPIs prediction [8]. A possible explanation is that the hybrid approach must deal with a high number of samples for model training (37× compared to the direct DNN in this study), and also the number of trainable DNN model parameters is around 16 million. Therefore, the hybrid approach requires higher computational power compared to DNN for the direct KPIs prediction. Also, the time to estimate the KPIs in the hybrid approach is increased from milliseconds to seconds due to the post-processing tool. However, the computational time remains much lower than the time for a FE simulation, which takes around 3 – 5 hours per machine design on a single-core CPU.

V. CONCLUSION

In this paper, we presented a physics and data-driven hybrid method for the performance analysis of the electrical machine. The proposed multi-branch DNN is trained to predict the intermediate measures using supervised learning. This hybrid approach is better than the existing parameter-based direct DL approach in terms of KPIs estimation and flexibility. This gain can be explained by two facts. Firstly, learning a few time-steps of intermediate measures is expected to be simpler than multiple (possibly independent) cross-domain KPIs and secondly, the post-processing tool exploits the laws of physics and thus ensures that the KPIs fulfill the right constraints. Also on the application side, this approach makes the calculation of KPIs independent from the system parameters by predicting intermediate measures during the optimization and it enables the analysis of electrical machines with the calculation of more complex performance measures. We have demonstrated that the trained multi-branch DNN meta-model evaluates new designs at a much lower computational effort than the FE simulation. In future work, the proposed hybrid approach can be applied to many query scenarios, e.g. uncertainty quantification, sensitivity analysis, and multi-objective optimization.

REFERENCES

[1] P. D. Barba, Multiobjective Shape Design in Electricity and Magnetism, (Lecture Notes in Electrical Engineering Series). Berlin, Germany: Springer, 2010.
[2] S. Koziel and L. Leifsson, Surrogate-Based Modeling and Optimization: Applications in Engineering, Berlin, Germany: Springer, 2013.
[3] Y.-B. Yan et al., “Torque estimation and control of pmsm based on deep learning,” in Proc. IEEE 22nd Int. Conf. Elect. Mach. Syst., 2019, pp. 1–6.
[4] H. Kurtović and I. Hahn, ”Neural network meta-modeling and optimization of flux switching machines,” in Proc. IEEE Int. Electric Mach. Drives Conf., 2019, pp. 629–636.
[5] A. Khan, M. H. Mohammadi, V. Ghorbanian, and D. Lowther, “Efficiency map prediction of motor drives using deep learning,” IEEE Trans. Magn., vol. 56, no. 3, pp. 1–4, Mar. 2020, Art. no. 7511504.
