A Semi-Supervised Method for Key Performance Indicator Prediction of Electrical Machines under a Self-Training Framework

Nuo Chen
Electrotechnical Institute
Karlsruhe Institute of Technology
Karlsruhe, Germany
nuo.chen@kit.edu

Christian Digel
Electrotechnical Institute
Karlsruhe Institute of Technology
Karlsruhe, Germany
christian.digel@kit.edu

Martin Doppelbauer
Electrotechnical Institute
Karlsruhe Institute of Technology
Karlsruhe, Germany
martin.doppelbauer@kit.edu

***Abstract*—Machine learning metamodels have been frequently used for predicting key performance indicators (KPIs) of electrical machines because of their rapid and precise predicting capabilities. However, the conventional procedure of data collection using finite element analysis is time-consuming. To shorten data collection time and improve the prediction accuracy of metamodels, we propose a semi-supervised learning approach using self-training techniques. Initially, we utilize the labeled samples to train a geometry classifier and a KPI predictor. We employ the classifier to eliminate geometrically infeasible designs from unlabeled data. For the remaining feasible designs, the predictor is applied to predict their KPIs. Since not all predictions of the trained metamodels are accurate and reliable, we introduce a confidence score using Monte-Carlo dropout to obtain trustworthy predictions as pseudo-labels for unlabeled data. Finally, the generated pseudo-labels and original labeled data are jointly used for metamodel retraining. The results show that the original dataset can be iteratively expanded in a short period of time, and the performance of both metamodels is improved.**

***Index Terms*—electrical machine, finite element analysis, probabilistic machine learning, Monte-Carlo dropout, semi-supervised learning, self-training.**

I. Introduction

With the growth of the electric vehicle industry, the electrical machine, which serves as one of the core parts of the drive chain, has gained more attention [1]. Permanent magnet synchronous machines (PMSMs) are frequently utilized because of their excellent characteristics, including high efficiency, high power density, and compact size [2], [3]. However, optimizing PMSM has the challenge of lengthy computation time caused by applying finite element analysis (FEA). Speeding up the optimization process has become a key area of current study. Hence, researchers are increasingly directing their attention to metamodel-based optimization approaches [4]. The machine learning technique has become popular among many metamodels for its high prediction accuracy and rapid computation speed [5]. In [6], a convolutional neural network (CNN) was trained to predict torque using a large amount of cross-sectional images of a PMSM as input. This method greatly decreased the computing cost of topology optimization while ensuring the optimization's reliability. In [7], the CNNs were implemented combined with transfer learning to handle small datasets and further accelerate the optimization process. Since the dimensions of images as input are large, the study [8] proposed to use simplified finite element models to reduce computation time. To predict efficiency maps of electrical machines, an approach using a CNN and a recurrent neural network (RNN) was introduced [9]. The CNN was adopted for flux linkage calculation, and the RNN was trained to predict a sequence of efficiency values. Another work [10] provided a comprehensive comparison between parameter-based and image-based metamodels. The results showed that parameter-based neural networks (NNs) possessed better prediction accuracy and less computational burden, while image-based NNs were better able to handle reparametrization. Moreover, in [11], different machine learning metamodels were employed for high-speed electrical machine optimization considering multiphysics coupling characteristics. During the optimization, a geometry classifier was used to filter out invalid design solutions. At present, an automatic design system for interior PMSMs was proposed [12]. With the help of a generative adversarial network, numerous rotor images were generated, which were later fed to a trained CNN to predict magnet flux linkage and d- and q-axis inductances.

So far, all of the above studies are based on supervised learning [13]. It requires a large quantity of well-labeled samples to train the metamodel properly in order to obtain good generalization ability. Nevertheless, in the field of design optimization for electrical machines, if a large dataset is desired for metamodels, the time required to collect samples using FEA is substantial. To solve this issue, we proposed a semi-supervised method for generating more data from a relatively small dataset obtained by FEA. Semi-supervised learning is a recent type of machine learning technique developed over the past decade. The idea aims to address the problem of reduced performance in conventional supervised learning due to a lack of training data by using unlabeled data during model training with limited labeled data [14]. It has been widely used in various fields of research, such as object detection [15], fault

diagnosis [16], medical diagnosis [17], and power engineering [18]. Self-training is an important learning technique in semi-supervised learning [19]. The main idea is to find a way to augment the labeled dataset with unlabeled datasets. In [20], a self-training framework was provided to iteratively produce data samples for expanding the training set. The classifier was then retrained after every iteration, and its performance was improving progressively.

In this study, we present a self-training framework utilizing semi-supervised learning to label unlabeled data to expedite the sampling procedure. In the beginning, a classifier (geometry checker) and a regression model (predictor) are trained using initial data labeled by FEA. This classifier examines design feasibility, while the regression model predicts the key performance indicators (KPIs) of the feasible designs, such as maximum torque and maximum power. Next, pseudo-labels are created by applying a confidence score to filter out certain predictions made by the trained models on unlabeled data, as not all predictions are expected to be trusted. The probabilistic metamodel, namely the Monte-Carlo (MC) dropout [21], is introduced to offer uncertainty information. Subsequently, the filtered pseudo-labels are integrated with the initial labeled data, and retraining is conducted on the combined dataset. The entire process can be repeated many times until convergence is reached.

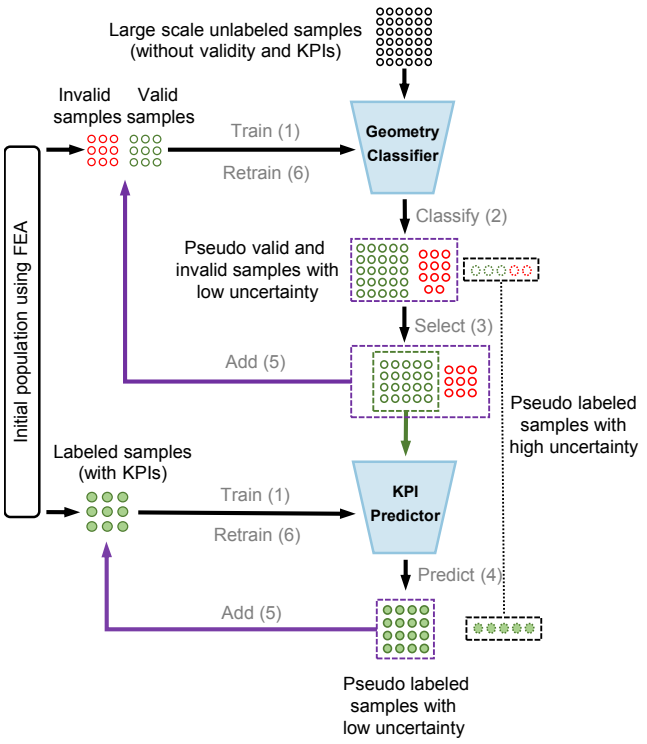

Fig. 1. Proposed self-training framework based on semi-supervised learning.

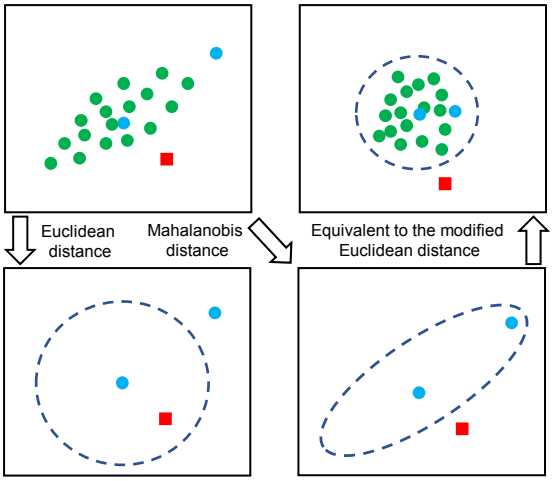

Fig. 2. Difference between Mahalanobis distance and Euclidean distance.

II. Framework and Dataset

A. Self-training Framework

The proposed self-training framework is illustrated in Fig. 1. First of all, Latin hypercube sampling (LHS) was utilized to generate sample vectors due to its efficient coverage of design parameter space. The samples were then identified as geometrically valid and invalid designs. Valid designs were calculated via FEA to obtain corresponding KPIs. All of the above was accomplished through the simulation toolchain we have developed. At this point, we had two initial datasets: one containing only labels about whether the design was valid, was used to train the classifier, and the other with simulated KPIs was used to train the predictor that predicted KPIs for valid samples. Subsequently, the detailed steps are described as follows:

1) *Train*: We utilized the initial dataset to train a geometry checker for the first time. The other dataset with simulated KPIs was fed to a regression metamodel to predict KPIs.
2) *Classify*: A large amount of data was generated via LHS. The geometry checker was then employed to separate these data into pseudo valid and invalid ones. Further categorization was done based on the confidence scores from probabilistic classifier. We discarded the data with high uncertainty.
3) *Select*: Due to the fact that the initially trained classifier had a certain number of classification errors, we applied the Mahalanobis distance [22] to calculate the similarity of the data with pseudo-labels to the initial dataset in order to remove the outliers from the data. The Mahalanobis distance can be seen as a modification of the Euclidean distance, correcting the problem that the dimensions are scale-inconsistent and correlated, as shown in Fig. 2. It can be expressed as follows:

$$D_M = \sqrt{(x_i - x_j)S^{-1}(x_i - x_j)}, \tag{1}$$

where x_i is the data point with pseudo-label and x_j is the data point from the initial dataset. S^{-1} describes the covariance matrix for initial labeled samples.

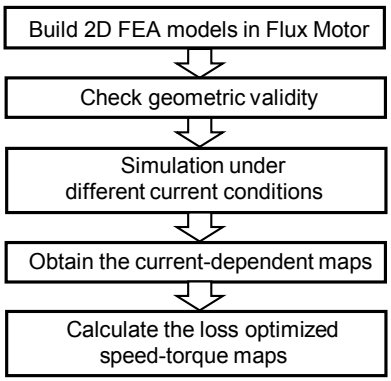

Fig. 3. Workflow of simulation toolchain.

4) *Predict*: The trained predictor was then adopted to predict KPIs for the selected pseudo valid data. Up until this point, we obtained new data with pseudo-labels regarding predicted KPIs.
5) *Add*: Following the selection process, pseudo valid and invalid data were incorporated into an initial dataset. In addition, the data with predicted KPIs were added to the other initial dataset.
6) *Retrain*: With the help of this new integrated dataset, we retrained the classifier and predictor to achieve better prediction performances.

This approach can be repeated iteratively until the metamodel's performance no longer improves with new joint dataset.

B. Initial Dataset Generation

In this study, the initial datasets were obtained through an in-house automatic simulation toolchain. The workflow of this simulation process is illustrated in Fig. 3. In the beginning, the design parameters generated by LHS were integrated with the machine template to construct 2D models in the commercial software Altair FLUX. After eliminating all the invalid designs, FEA was conducted on the remaining designs based on 30 operating points (combination of armature current I_a and current phase angle β). In the post-processing stage, we obtained the current-dependent maps by interpolation. A loss-based optimization was then performed to achieve speed-torque maps. Finally, the KPIs for each design were extracted.

Our machine template was a PMSM with V-magnets, as shown in Fig. 4. A total of 13 input design parameters and four KPIs were included in the dataset. Table I shows the parameters about the reference machine. The initial training set for the classifier included 3002 samples (1637 valid designs, 1365 invalid designs), while the initial training set for the KPI predictor contained 1473 samples. The number of samples in two test sets was 166 and 82, respectively.

III. Training of Neural Networks and Results

A. MC Dropout

Unlike traditional deterministic (point) prediction, probabilistic metamodels provide additional information on prediction uncertainty along with point prediction. In [23], it has been proven that a neural network with dropout layers before each hidden layer is mathematically close to a Gaussian process. Below is the key for implementing the MC dropout method: Add a dropout layer with a suitable dropout rate before each hidden layer, no matter if it is the first or last hidden layer. The work [21] suggests using l_2 regularization when the objective is to enhance uncertainty as it moves away from the data. It is also important to note that the dropout layers in a traditional NN are activated during training and deactivated during prediction. However, for a probabilistic NN using MC dropout, the dropout layers remain active during both the training and prediction phases, which means that

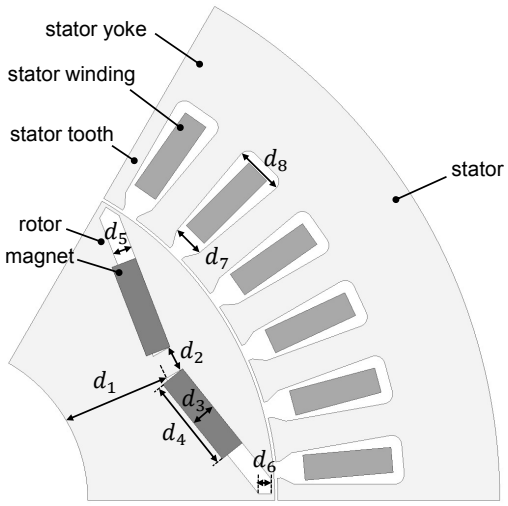

Fig. 4. Reference PMSM geometry with partial design parameters.

TABLE I
Details of Reference Machine Parameters

Symbol	Design parameter	Min.	Max.	Unit
d_1	Distance from duct bottom to shift	2.0	24.0	mm
d_2	Duct opening width	1.0	5.0	mm
d_3	Magnet thickness	3.0	6.5	mm
d_4	Magnet width	26.0	48.0	mm
d_5	Duct thickness	2.0	5.5	mm
d_6	Rib height	0.5	2.0	mm
d_7	Slot wedge maximum width	3.8	7.0	mm
d_8	Slot body bottom width	6.6	9.8	mm
d_9	Rib width	1.0	15.0	mm
d_{10}	Minimum distance between side magnets	1.0	22.0	mm
d_{11}	Slot opening width	1.8	3.2	mm
d_{12}	Slot opening height	0.3	1.6	mm
d_{13}	Slot wedge height	1.5	4.2	mm
	Constant parameter	**Value**		
c_1	Number of poles	6		-
c_2	Number of stator slots	36		-
c_3	Inverter voltage	650.0		V
c_4	Inverter current (RMS)	300.0		A
	Indicator			
k_1	Maximum torque			Nm
k_2	Maximum power			kW
k_3	Torque ripple at maximum torque			Nm
k_4	Efficiency at corner point			%

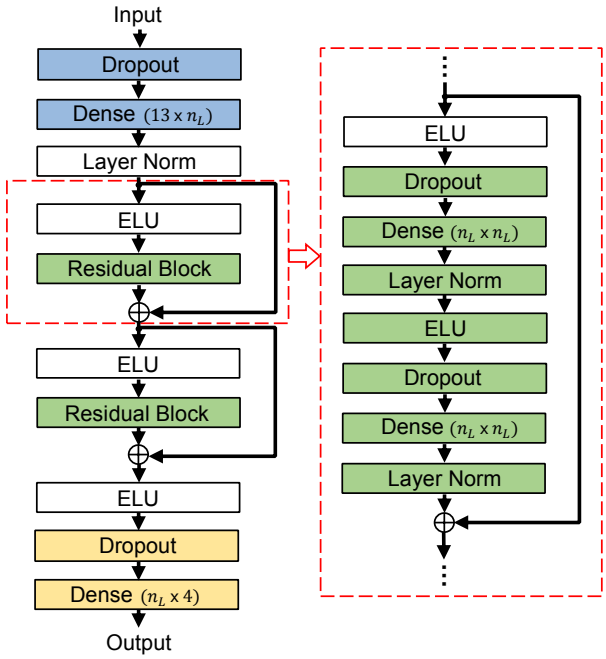

Fig. 5. MC dropout neural network architecture with two residual blocks.

each forward pass yields a slightly different prediction. The mean and standard deviation of n times prediction values can be considered as the final point estimation and uncertainty quantification, respectively [24].

B. NN Architecture and Training

In our work, we employed deep NNs with residual blocks [25] for both the classifier and regression model. Fig. 5 shows the architecture of the NN with two residual blocks. As the depth of the network continues to increase, model training becomes difficult, and the network degrades. Therefore, this issue can be well solved by adding residual blocks [25]. Each dense layer began with a dropout layer. We also applied layer normalization after each dense layer and before the activation function to enhance the stability and convergence speed of the NN [26].

All the data from initial datasets were preprocessed using z-score normalization to eliminate the effect of scale between different feature data [27], which can be described as follows:

$$x' = \frac{x - \mu}{\sigma}, \tag{2}$$

where μ is the mean of data, and σ is the standard deviation of data. The performance of NNs is greatly influenced by the selection of hyperparameters, such as learning rate, the quantity of neurons per dense layer, optimizer, and so on. Furthermore, since we introduced dropout layers and residual blocks, additional hyperparameters namely dropout rate and the number of residual blocks, need to be determined before training. To lessen the computational burden, trial and error was performed at the beginning to reduce the range of each hyperparameter. The random search optimization was then executed to determine the final hyperparameter values. The 5-fold cross-validation was utilized to ensure reliability. In addition, we also applied early stopping techniques during the training process to avoid overfitting. The determined hyperparameters are summarized in Table II.

TABLE II
DETAILS OF HYPERPARAMETERS

Hyperparameter	Optimized values	
	Classifier	Predictor
Learning rate	$1.0 \cdot 10^{-2}$	$0.4 \cdot 10^{-2}$
Dropout rate	$1.0 \cdot 10^{-1}$	$0.5 \cdot 10^{-1}$
Activation function	ELU	ELU
Number of neurons	32	106
Number of residual blocks	1	2
Optimizer	Adam	Adam

TABLE III
CONFUSION MATRIX

		Predicted	
		1	0
Actual	1	TP	FN
	0	FP	TN

In order to evaluate the prediction accuracy of KPI predictor, the evaluation metrics, namely symmetric mean absolute percentage error (SMAPE) and coefficient of determination (R^2) were applied. SMAPE is a correction for mean absolute percentage error, which better avoids the problem of the calculation being too large due to small actual values. R^2 measures how well the model approximates the actual data. They are expressed as below:

$$SMAPE = \frac{100\%}{N} \sum_{i=1}^{N} \frac{|y_i - y_i^*|}{(|y_i| + |y_i^*|)/2}, \tag{3}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - y_i^*)^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2}, \tag{4}$$

where N represents the number of test data. y_i^* describes the prediction value and $\bar{y}$ is the average true value. To assess the degree of uncertainty, the prediction interval normalized average width (PINAW) was introduced, which assesses the quality of predictive intervals. It can be calculated by:

$$PINAW = \frac{1}{N \cdot R} \sum_{i=1}^{N} (y_i^{*U} - y_i^{*L}), \tag{5}$$

where R represents the range of point predictions. y_i^{*U} and y_i^{*L} are the predicted upper and lower bounds of the confidence interval, respectively. To evaluate the quality of classification, we calculated the true positive rate (TPR) and the false positive rate (FPR):

$$TPR = \frac{TP}{TP + FN}, \tag{6}$$

$$FPR = \frac{FP}{FP + TN}. \tag{7}$$

TABLE IV
EVALUATION SUMMARY OF KPI PREDICTOR BASED ON DIFFERENT DATASETS

Dataset	KPI	R^2	SMAPE	PINAW
Initial (1555 samples)	k_1	0.971	2.053	0.263
	k_2	0.963	3.209	0.259
	k_3	0.823	10.37	0.344
	k_4	0.585	0.569	0.279
Iteration 1 (2503 samples)	k_1	0.982	1.794	0.209
	k_2	0.971	2.879	0.192
	k_3	0.860	9.645	0.245
	k_4	0.614	0.539	0.226
Iteration 2 (4042 samples)	k_1	0.991	1.605	0.167
	k_2	0.980	2.713	0.157
	k_3	0.876	9.237	0.194
	k_4	0.652	0.537	0.155

The meaning of TP, FN, FP, and TN are demonstrated in Table III.

C. Training Results

After initially training the classifier and predictor, we utilized LHS to create 10000 candidate designs for pseudo-label generation. These 10000 candidates contained only the input design parameters, i.e., they did not have any simulation information. Moreover, they encompassed both feasible and infeasible designs. The geometry checker initially eliminated any infeasible designs. Next, the KPI predictor was used to forecast the remaining feasible designs. A total of 2852 data points with low uncertainty were obtained after passing through the classifier. The average of the largest and smallest standard deviations in the predictions was taken as the boundary to distinguish between low and high uncertainty. Subsequently, 162 samples were further filtered out using Mahalanobis distance. We obtained 2690 samples with valid or invalid pseudo-labels, which were added to the initial dataset for the classifier. Furthermore, the samples with valid pseudo-labels were then fed to the KPI predictor. Finally, 948 samples with low uncertainty were acquired with KPI pseudo-labels, which were also integrated into the initial dataset for the predictor. The prediction performance of the KPI predictor based on the initial dataset and the expanded dataset after the first iteration is demonstrated in Fig. 6. The commonly used 95% uncertainty intervals (UIs) were calculated. It can be seen that the prediction accuracy of the metamodel obtained by training on the extended dataset increased and the uncertainty decreased. The specific values of the evaluation metrics are shown in Table IV. Fig. 7 illustrates the receiver operating characteristic (ROC) curves [28] based on different datasets. The areas of the ROC curves, also known as area under curves (AUC) [29], were calculated to evaluate the performance of the classifier. In the case of the trained classifier based on the expanded dataset after iteration 1, the AUC was larger, which indicated better classification ability.

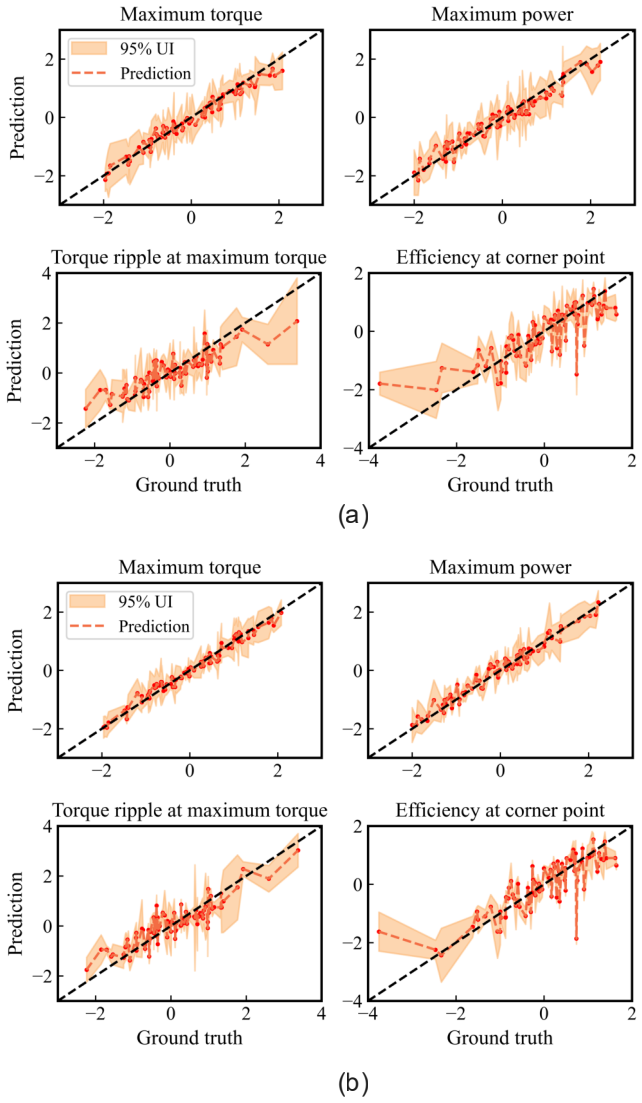

Fig. 6. Prediction performance of KPI predictor based on initial dataset and expanded dataset after first iteration. (a) Initial dataset. (b) Expanded dataset.

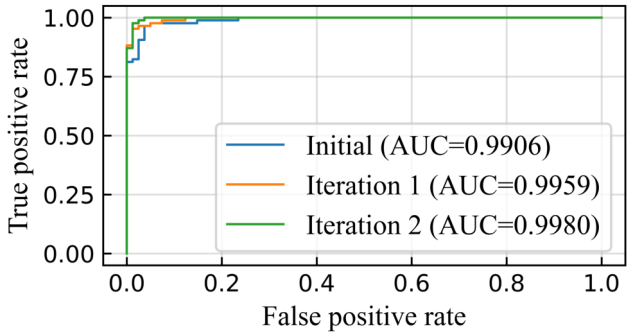

Fig. 7. ROC curves of geometry classifier.

We did a second iteration to further confirm the effectiveness of the self-training framework. The whole second iteration procedure mirrored the first iteration. Following the selection

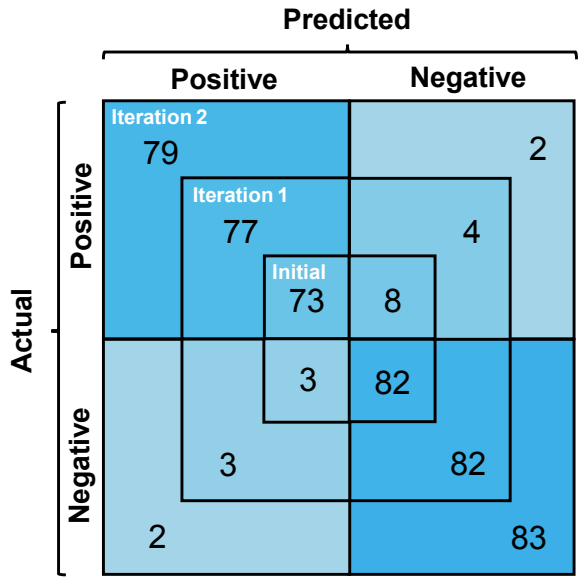

Fig. 8. Confusion matrix based on initial dataset and expanded datasets with threshold 0.5.

algorithm, we obtained 4248 samples with low uncertainty for the expanded classifier dataset. After the KPI prediction phase, 1539 samples with KPIs predicted by the metamodel were added to the dataset. Based on the expanded dataset after the second iteration, the classifier and the predictor were retrained once more. The detailed predictor performance is listed in Table IV. As the dataset size rose, the prediction accuracy of the metamodel improved. Furthermore, the associated predictive uncertainty diminished. The confusion matrix with specific threshold 0.5 is displayed in Fig. 8. In this case, the classifier trained based on the dataset obtained after the second iteration had the best classification performance.

D. Limitations

Our proposed self-training framework could create more training data through continuous iteration. This allowed us to update the metamodels with a growing dataset, leading to ongoing improvement in prediction accuracy. However, since the new data were obtained through model predictions, they deviated from the real data. We then added the new data to the initial dataset, which could contaminate the original dataset. As a result, we employed a two-step selection procedure to eliminate as many unreliable predictions as possible. This process involved initially assessing the predictions based on their uncertainty, followed by applying Mahalanobis distance to remove any remaining outliers. Nevertheless, it is impractical to completely ensure the elimination of all data that drastically deviates from the ground truth. Hence, once the extended dataset reaches a specific threshold of noise, the metamodel's performance will stop to increase and may even decline. Additionally, the original dataset has to fulfill certain requirements, namely, the initially trained metamodels need to possess a certain level of prediction accuracy. If the metamodel is initially inaccurate, the subsequent generated data will progressively diverge more and more from the true values.

IV. Conclusion

In this paper, we proposed a self-training framework based on semi-supervised learning to enlarge initial datasets. Two metamodels, namely the geometry classifier and KPI predictor, collaborated to produce pseudo-labels for unlabeled samples. We applied the probabilistic metamodel MC dropout to enhance the quality of the pseudo-labeled data. It not only offered point predictions but also provided the uncertainty associated with those predictions. The results showed that each iteration of this approach quickly produced pseudo-labeled samples to augment the original datasets. Increasing the dataset size enhanced the performance of the classifier and predictor. The uncertainty of the forecast outcomes was also reduced. In future work, this method will be combined with multi-objective optimization for various electrical machines. Furthermore, it would be an important research direction to partially replace FEA to generate sufficient data quickly.

References

[1] E. Agamloh, A. Jouanne, and A. Yokochi, "An overview of electric machine trends in modern electric vehicles," Machines, vol. 8, no. 2, p. 20, 2020.

[2] Y.-H. Jung, M.-R. Park, K.-O. Kim, J.-W. Chin, J.-P. Hong, and M.-S. Lim, "Design of high-speed multilayer IPMSM using ferrite PM for EV traction considering mechanical and electrical characteristics," IEEE Trans. Ind. Appl., vol. 57, no. 1, pp. 327–339, Jan. 2021.

[3] X. Liu, H. Chen, J. Zhao, and A. Belahcen, "Research on the Performances and Parameters of Interior PMSM Used for Electric Vehicles," IEEE Trans. Ind. Electron., vol. 63, no. 6, pp.3533–3545, 2016.

[4] J. Kudela and R. Matousek, "Recent advances and applications of surrogate models for finite element method computations: A review," Soft Computing, vol. 26, no. 24, pp.13709-13733, Jul. 2022.

[5] S. Zhao, F. Blaabjerg, and H. Wang, "An overview of artificial intelligence applications for power electronics," IEEE Transactions on Power Electronics, vol. 36, no. 4, pp. 4633–4658, 2021.

[6] H. Sasaki and H. Igarashi, "Topology optimization accelerated by deep learning," IEEE Transactions on Magnetics, vol. 55, no. 6, pp. 1–5, 2019.

[7] J. Asanuma, S. Doi and H. Igarashi, "Transfer Learning Through Deep Learning: Application to Topology Optimization of Electric Motor," IEEE Transactions on Magnetics, vol. 56, no. 3, pp. 1-4, March 2020.

[8] S. Barmada, N. Fontana, L. Sani, D. Thomopulos and M. Tucci, "Deep Learning and Reduced Models for Fast Optimization in Electromagnetics," IEEE Transactions on Magnetics, vol. 56, no. 3, pp. 1-4, March 2020.

[9] A. Khan, M. H. Mohammadi, V. Ghorbanian, and D. Lowther, "Efficiency map prediction of motor drives using deep learning," IEEE Transactions on Magnetics, vol. 56, no. 3, pp. 1–4, 2020.

[10] 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.

[11] J. Gu, W. Hua, W. Yu, Z. Zhang and H. Zhang, "Surrogate Model-Based Multiobjective Optimization of High-Speed PM Synchronous Machine: Construction and Comparison," IEEE Transactions on Transportation Electrification, vol. 9, no. 1, pp. 678-688, March 2023.

[12] 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 Transactions on Energy Conversion, vol. 38, no. 1, pp. 724-734, March 2023.

[13] J. Gui, et al. "A survey of self-supervised learning from multiple perspectives: Algorithms, theory, applications and future trends," ArXiv:2301.05712, 2023.

[14] J. E. Van Engelen and H. H. Holger, "A survey on semi-supervised learning," Machine learning, vol. 109, no. 2 pp. 373-440, 2020.

[15] Q. Zhou, C. Yu, Z. Wang, Q. Qian, and H. Li, "Instant-teaching: An end-to-end semi-supervised object detection framework," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4081-4090. 2021.
[16] S. Zhang, et al., "Semi-supervised bearing fault diagnosis and classification using variational autoencoder-based deep generative models," IEEE Sensors Journal, vol. 21, no. 5, pp. 6476-6486, 2020.
[17] N. Al-Azzam and I. Shatnawi, "Comparing supervised and semi-supervised machine learning models on diagnosing breast cancer," Annals of Medicine and Surgery, vol. 62, pp. 53-64, 2021.
[18] B. Zhou, et al., "Short-term prediction of wind power and its ramp events based on semi-supervised generative adversarial network," International Journal of Electrical Power & Energy Systems, p. 106411, 2021.
[19] M.-R. Amini, et al., "Self-training: A survey," ArXiv:2202.12040, 2022.
[20] F. Li, D. A. Clausi, L. Xu, and A. Wong, "ST-IRGS: A Region-Based Self-Training Algorithm Applied to Hyperspectral Image Classification and Segmentation," IEEE Transactions on Geoscience and Remote Sensing, vol. 56, no. 1, pp. 3-16, Jan. 2018.
[21] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," International Conference on Machine Learning, PMLR, Jun. 2016.
[22] M. T. Law, et al., "Closed-Form Training of Mahalanobis Distance for Supervised Clustering," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp: 3909-3917, 2016.
[23] A. Damianou and N.D. Lawrence, "Deep gaussian processes," Artificial Intelligence and Statistics, pp. 207–215, 2013.
[24] X. Wu, P. Wagner, and M. F. Huber, "Quantification of Uncertainties in Neural Networks," Springer International Publishing, pp. 276–287, 2023.
[25] K. He, et al., "Deep residual learning for image recognition," Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.
[26] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," ArXiv:1607.06450, 2016.
[27] S. G. K. Patro, K. K. Sahu, "Normalization: A preprocessing stage," ArXiv:1503.06462, 2015.
[28] V. Bewick, L. Cheek, and J. Ball, "Statistics review 13: receiver operating characteristic curves," Critical Care, vol. 8, no. 6, pp. 1-5, 2004.
[29] A. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," Pattern Recognition, vol. 30, no. 7, pp. 1145-1159, 1997.

V. BIOGRAPHIES

Nuo Chen received the bachelor's degree in automotive engineering from Harbin Institute of Technology, China, in 2018, and the master's degree in autonomous systems from the University of Stuttgart, Germany, in 2022. He is currently working toward the Ph.D. degree at the Institute of Electrical Engineering, Karlsruhe Institute of Technology, Germany. His research interests include machine learning, deep learning, and the optimization design of electrical machines.

Christian Digel obtained his B.Sc. and M.Sc. degree in Industrial Engineering from the Karlsruhe Institute of Technology (KIT) and TU Delft. With the end of his studies in 2021 he joined the Electrotechnical Institute (ETI) at KIT as a research associate and PhD student. His research focus is on parametric and nonparametric multi-physics optimization and surrogate modeling in the design of electrical machines.

Martin Doppelbauer was born in Altenhundem, Germany. He received the Dipl.-Ing. and Dr.-Ing. degrees in electrical engineering from the University of Dortmund, Germany in 1990 and 1995, respectively. He was with the companies Danfoss Bauer in Esslingen and SEW Eurodrive in Bruchsal, Germany, as Senior Manager for electric motor development. Since 2011 he is full professor at the Institute of Electrical Engineering (ETI) at the KIT. He holds a chair for Hybrid Electric Vehicles. Martin Doppelbauer is also active in national and international standardization and the chairman of IEC Technical Committee 2 Rotating Electrical Machines.