An Effective Ensemble-Based Method for Creating On-the-Fly Surrogate Fitness Functions for Multi-Objective Evolutionary Algorithms

Alexandru-Ciprian Zăvoianu*‡, Edwin Lughofer*, Gerd Bramerdorfer†‡ Wolfgang Amrhein†‡ and Erich Peter Klement*‡
*Department of Knowledge-based Mathematical Systems / Fuzzy Logic Laboratory Linz-Hagenberg,
Johannes Kepler University of Linz, Austria
†Institute for Electrical Drives and Power Electronics, Johannes Kepler University of Linz, Austria
‡ACCM, Austrian Center of Competence in Mechatronics, Linz, Austria

***Abstract*—The task of designing electrical drives is a multi-objective optimization problem (MOOP) that remains very slow even when using state-of-the-art approaches like particle swarm optimization and evolutionary algorithms because the fitness function used to assess the quality of a proposed design is based on time-intensive finite element (FE) simulations. One straightforward solution is to replace the original FE-based fitness function with a much faster-to-evaluate surrogate. In our particular case each optimization scenario poses rather unique challenges (i.e., goals and constraints) and the surrogate models need to be constructed on-the-fly, automatically, during the run of the evolutionary algorithm. In the present research, using three industrial MOOPs, we investigated several approaches for creating such surrogate models and discovered that a strategy that uses ensembles of multi-layer perceptron neural networks and Pareto-trimmed training sets is able to produce very high-quality surrogate models in a relatively short time interval.**

Keywords*—multi-objective evolutionary algorithms, surrogate fitness evaluation, artificial neural networks, ensemble regression models

I. Introduction and State of the Art

The task of designing electrical drives is a multi-objective optimization problem / process (MOOP) that aims to find machines that simultaneously have: a high efficiency, very good fault tolerance, easy to control operating characteristics, and, last but not least, a low manufacturing price. Like with most MOOPs, such problems rarely have one single solution. Solving them requires finding a set of non-dominated solutions called the *Pareto-optimal set*. Each solution in this set is better than any other solution in the set with regards to at least one optimization objective (i.e., it is not fully dominated by another solution).

In the past, the problem of optimizing the design of an electrical drive was solved by performing a parameter sweep (grid search) and calculating a maximum of several hundred designs [1]. Calculating a design actually means predicting the operational behavior of the machine under certain conditions and, because of the non-linear behavior of the materials involved, for each design, such a prediction is usually based on the results obtained after performing several time intensive (3 to 10 minutes) finite element (FE) simulations. In spite of using modern (population-based) optimization approaches like genetic algorithms [2] and particle swarm optimization [3], the present day design process is still very slow (one complete optimization run can take several days) even when distributing the computations over a computer cluster. This is because these optimization methods also use FE simulations in order to evaluate the performance (fitness) of an individual (design).

A straightforward solution aimed at reducing the run time of an evolutionary algorithm that has a very time-intensive fitness function is to approximate the original fitness function with *surrogate models* [4]. In our case, this means substituting the time-intensive fitness functions based on FE simulations with very-fast-to-evaluate surrogates based on accurate regression models. The surrogate models will act as direct mappings between the design parameters of the electrical drives (inputs) and the objective and constraint values that are normally estimated via FE (outputs). Throughout this paper we shall refer to the objectives and constraints that we wish to model using surrogates as *targets*. Four very well documented general overviews on surrogate based analysis and optimization can be found in [5], [6], [7] and [8]. In the field of electrical drive design optimization, surrogate-based approaches are described in [9] and [10].

As every electrical drive design MOOP we wish to solve has quite specific characteristics (design parameters, objectives and constraints), an important requirement is that the surrogate modeling is performed on-the-fly (during the run of the optimization). Moreover, the whole surrogate construction stage should be fully automatic (not require any human interaction). For performing the optimizations we are using multi-objective evolutionary algorithms [11] and, therefore, the data sets used for constructing the surrogates will contain (part of) the FE-evaluated individuals computed in the first N "generations" of the evolutionary process.

The goal of the present research is to describe an *effective ensemble-based method for creating on-the-fly surrogate fitness functions*. The focus is on analyzing how different strategies for constructing surrogates for non-linear targets compare in terms of *modeling accuracy*, *required training time*, and *prediction stability* when considering three industrial optimization scenarios from the field of electrical drive design.

II. Constructing On-the-Fly Surrogate Fitness Functions

A. General Considerations

There are at least three important general decisions that need to be made when constructing on-the-fly surrogate fitness functions. The first one is regarding the structure of the surrogate fitness function. What is to be preferred: a surrogate fitness function that is using one single model to predict all the elicited targets (e.g., a neural network with multiple output nodes), or a surrogate fitness function that relies on several surrogate models as each target is modeled independently? In our optimization scenarios, as the *targets are usually highly conflicting* (i.e., price vs. performance vs. reliability) and *exhibiting different degrees of non-linearity*, we have chosen to *create a separate surrogate model for each target* and we propose a two tier modeling process. In the first stage we attempt to construct a linear model for all the considered targets. If an obtained linear model does not display a good prediction performance (the value of the coefficient of determination - R^2 when applying the model on the training set is smaller than 0.925), in a second stage, we attempt to create a non-linear model for the associated target.

The next important general decision concerns the structure of the data set that will be used in order to train the surrogate models. On the one hand, a training set that is too small or not well distributed over the search space is likely to produce surrogate models that do not generalize well. On the other hand, a training set that is too large is likely to incur longer surrogate model training times.

In [12], the surrogate model construction stage was started after the completion of $N = 25$ generations as, for the considered MOOPs and the considered model training and model selection methods, the data sets produced until this stage of the optimization process were sufficient for training high-quality surrogate models that maintained their very good prediction accuracy for the next 75 generations. After experimenting with several other electrical drive optimization scenarios, we have reached the conclusion that, without extensive apriori knowledge about the particularities of the targets to be modeled, $N = 25$ is a generally good threshold for starting the surrogate model construction stage for non-linear targets. In light of these observations, all the comparative results presented in Section IV are based on training sets constructed with the $N = 25$ setting. Choosing this particular setting for our experiments also serves to create a comparison basis, as the present work essentially describes our efforts to improve (especially in terms of required training time) the surrogate model construction mechanism presented in [12].

The third important decision is related to the method used to create surrogate models for the non-linear targets. The methods that we considered are artificial neural networks (more precisely multi-layered perceptrons - MLPs with one hidden layer), support vector regression (SVR), and radial basis function (RBF) networks. Our choice for these methods is generally motivated by the fact that they are regarded as powerful modeling tools (i.e., they possess the universal approximation capability). Particularly, these methods have been successfully used on several occasions for surrogate modeling purposes [13] [14]. Modeling the non-linear targets is by far the most challenging part (both accuracy and time-wise) of the surrogate fitness function construction process and in the next sections we shall describe and analyze several attempts of tackling this task.

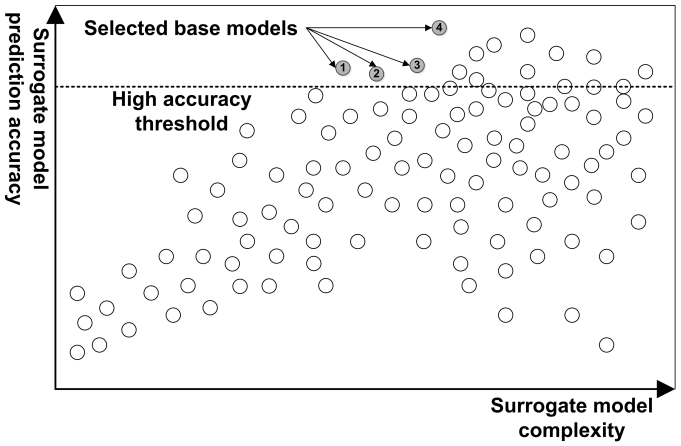

Fig. 1. Diagram of the $ENS^{thr(P)}(S), S > 3$ model selection scheme. The numbers indicate the order in which the base models are added into the ensemble predictor.

B. Strategies for Constructing Surrogate Models for Non-Linear Targets

We start the surrogate model construction stage after computing N initial generations of FE-evaluated individuals and we consider three types of training sets:

1) $full(N)$ - a training set that contains all the error-free individuals found during the first N generations of the optimization process.
2) $rand(C,N)$ - a training set that contains at most C samples randomly extracted from $full(N)$
3) $trim(C,N)$ - a training set that contains at most C samples extracted from $full(N)$ using the following strategy: At first, given an optimization scenario with c_1 targets to model, we extract a maximum of c_1 samples that are associated with the worst results of the elicited targets of this scenario. Secondly, we compute the Pareto non-dominated solution set from $full(N)$ and, if the size of this front is smaller than $c_2 = C - c_1$, we extract all the samples that correspond to the non-dominated solutions and add them to $trim(C,N)$. Finally, we randomly extract at most $c_3 = C - c_1 - c_2$ solutions from the remaining samples in $full(N)$ and also add them to the trimmed training set. If the size of the Pareto non-dominated solution set extracted from $full(N)$ is larger than c_2, we apply a Pareto/crowding-based selection method (i.e., non-dominated sorting [15] or environmental selection [16]) to select exactly c_2 individuals and add the associated samples to $trim(C,N)$.

As we are using parametric non-linear modeling methods, in order to select the best parameter configuration that will be used for training the surrogate model/models, we perform a grid search and select the best candidate based on 10-fold cross-validation performance. The final surrogate model is trained using the entire training set and the parameterization of

the previously computed best performing candidate. The main performance indicator is (constructed around) the coefficient of determination but sometimes we also use information related to the (structural) complexity of the resulting models. Given the results of a best parameter grid search, we are considering the following automatic model selection schemes:

1) SIN^{best} - selects the single best performing surrogate model when considering the averaged R^2 value over the cross-validation folds.
2) $SIN^{thr(P)}$ - selects a single surrogate model according to the selection method described in [12]: the best performing model is chosen by first computing a high accuracy threshold using the most accurate $P\%$ models and then selecting the least complex model (e.g., lowest number of hidden units in the case of MLPs) that has a prediction accuracy that is higher than this threshold. The accuracy assigned to a surrogate model is the mean value of R^2 minus the standard deviation of R^2 over the cross-validation folds.
3) $ENS^{best}(S)$ - selects S base models that will form an ensemble surrogate predictor. The individual models are chosen in decreasing order of the averaged R^2 value over the cross-validation folds. The final prediction of the ensemble is a simple average over the predictions of the individual models. Trying to see if such a basic averaging ensemble is able to achieve a higher prediction accuracy than the single surrogate models would make sense because there is virtually no extra computational cost associated with the method as all the individual base models are being created during the best parameter grid search.
4) $ENS^{thr(P)}(S)$ - also selects S base models that will form a simple averaging ensemble predictor. The individual models are chosen by extending the previously mentioned high accuracy threshold method. As such, after computing the threshold, models that exhibit a good accuracy vs. complexity trade-off are selected iteratively until the model count limit (i.e., S) is reached or until the model with the highest prediction accuracy is selected. In case of the latter, the ensemble structure is filled by applying a circular (round-robin) copy strategy on the individual models selected so far. When considering the scenario from Figure 1, where the base model labeled "4" has the highest prediction accuracy, if our goal would be to create an ensemble of 10 base models (i.e., $S = 10$), in the final surrogate predictor, the models labeled "1" and "2" would appear three times each and the models labeled "3" and "4" would appear twice. In the extreme case where only one model has a prediction accuracy higher than the threshold, the ensemble surrogate will consist of ten copies of this base model and will perform identical to the single-model surrogate that would be created by applying the $SIN^{thr(P)}$ selection scheme on the same best parameter grid search results.

It should be noted that all four model selection schemes are suitable for the three considered non-linear modeling methods. These modeling methods in turn, can be combined with any of the three types of training sets. Throughout the remainder of this paper we shall refer to the combination of modeling method + model selection scheme + training set as a *surrogate model construction strategy*. For example, $MLP-ENS^{thr(5)}_{trim(250,25)}(10)$ denotes a surrogate model construction strategy that aims to create an ensemble predictor that contains 10 base MLP models selected according to the high accuracy threshold method (threshold is computed using the most accurate 5% models) and trained using a Pareto-trimmed data set of no more than 250 samples that are selected from all the valid designs evaluated in the first 25 generations of the optimization process.

III. Experimental Setup

A. The Data Sets and the Elicited Targets

The data sets we are using in order to compare the different surrogate model creation strategies were obtained by performing optimization runs of 100 generations on three electrical drive design problems and by storing every generated (evolved) *valid individual* (design parameter vector) and its FE-computed associated target values.

All three optimizations were performed using the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [16] with standard genetic operators (SBX crossover [17] and polynomial mutation [18]) and a population size of 50. We also used a standard parameterization with a crossover probability of 0.9, a crossover distribution index of 20, a mutation probability of $1/|D|$ (where D is the design parameter vector) and a mutation distribution index of 20.

TABLE I. Information regarding the total number of valid designs generated during the optimization runs

Scenario	$N = 25$	$N = 50$	$N = 75$	$N = 100$
1	1206	2430	3605	4732
2	897	1855	2743	3622
3	799	1745	2685	3582

As we used a population size of 50, during each optimization run we evaluated 5000 electrical drive designs. Because some of the evolved design configurations were geometrically unfeasible or invalid with regards to given optimization constraints, the actual size of the data sets obtained after the completion of generation N is smaller than $N*50$. The number of valid individuals generated during the optimization runs for each optimization scenario after 25, 50, 75 and 100 generations is presented in Table I.

For all experiments, when constructing the Pareto-trimmed surrogate model training sets (i.e., $trim(C,N)$), if required, we applied the environmental selection strategy [16].

The first optimization scenario deals with an electrical motor with embedded magnets and has a design parameter vector of size six. There are five targets (three objectives and two constraints) that need to be estimated using the surrogate fitness function. Three of these targets, the negated efficiency (T1), the peak-to-peak-value of the motor torque for no current excitation (T2), and the peak-to-peak-value of the motor torque at load operation (T3) have been labeled as non-linear because

linear regression only achieved training R^2 values of 0.895, 0.7622, and 0.7799.

The second MOOP concerns an electrical machine featuring an exterior rotor. The design parameter vector contains seven geometric dimensions. Among the three targets (two objectives and one constraint) for which a surrogate model must be constructed, one (the losses of the system at load operation -T4) has been labeled as non-linear (linear regression training R^2 value of 0.9104).

The third optimization scenario also deals with a motor with an exterior rotor. The design parameter vector has a size of ten and two of the four targets (objectives) to be modeled are non-linear. These targets are the ohmic losses in the stator coils (T5) and the total losses due to material hysteresis and eddy currents in the ferromagnetic parts of the motor (T6). Linear regression achieved training R^2 values of 0.9156 and 0.7230.

B. The Best Parameter Grid Searches

The MLP, SVR and RBF networks implementations we used for our tests are largely based on the ones provided by the WEKA open source machine learning platform [19].

In the case of the MLP surrogates, training is done via standard backpropagation [20]. In the grid searches that we performed in order to find the best parameter configurations:

- the number of hidden units was varied between 2 and double the number of design variables;
- the learning rate (η) was varied between 0.05 and 0.40 with a step size of 0.05;
- the momentum parameter (α) was varied between 0.0 and 0.7 with a step size of 0.1;

This means that during the MLP grid search we constructed: 704 surrogate models for the targets T1, T2, and T3; 832 surrogate models for the target T4; 1216 surrogate models for the targets T5 and T6.

When training the MLP surrogate models we used an early stopping mechanism that terminated the training whenever the prediction error computed over a validation subset did not improve over 200 consecutive iterations. For every surrogate training task, this validation subset was constructed at the beginning of the actual training process by randomly sampling 20% of the instances from the initial training set (i.e., the one obtained after merging 9 out of 10 cross-validation folds). Whenever the early stopping criterion was not reached, the training process stopped after 10000 iterations.

In the case of SVR, we trained 630 surrogate models for each target as we varied: the general complexity parameter C between $[2^{-4}, 2^{-3}, ..., 2^{9}]$, the RBF kernel parameter γ between $[2^{-3}, 2^{-2}, ..., 2^{5}]$ and the ε parameter of the ε-intensive loss function between $[0.005, 0.01, 0.025, 0.05, 0.1]$.

For RBF networks, we trained 513 surrogate models for each target by varying the number of clusters between $[1, ..., 5, 10, 15, ..., 50, 60, ..., 100, 125, ..., 200, 250, 300, 400, 500]$ and the allowed minimum standard deviation for the clusters between $[0.05, 0.1, 0.50, 1.00, ..., 5, 6, ..., 10, 15, 20]$.

C. Surrogate Accuracy Evaluation

In order to estimate the quality of a given surrogate model created using the FE-evaluated individuals from the first $N = 25$ generations, we compute the overall and the generational R^2 values obtained when trying to predict the target values of the FE-evaluated individuals from the remaining 75 generations (i.e., unseen test data).

Computing the generational R^2 values helps us to estimate if the obtained surrogate model is able to maintain its initial accuracy level over the entire run of the optimization process. Stability is a highly desirable characteristic for the surrogate models as it ensures that no / very few surrogate re-training phases will be required during the optimization run.

IV. Results

All the tests that we report on have been executed on the same machine (8-core CPU, 4GB of RAM) in a multi-threaded environment that used a maximum of 6 threads. Surrogate model training times have been measured independently.

A. Single-Model Surrogates

In the first series of tests we wanted to check which of the three non-linear modeling methods produces the best single-model surrogates when training using all the valid samples from the first 25 generations (i.e., the $full(25)$ training set). The results are presented in Table II and indicate that MLPs perform better than SVR and RBF networks (linear regression results are also provided in order to offer more insight into the characteristics of the modeled targets). Furthermore, when considering the average performance over all elicited targets, the model selection method based on thresholding ($MLP - SIN^{thr(5)}$) displays a small accuracy improvement over simply selecting the best cross-validation model ($MLP - SIN^{best}$).

When also taking into account the average training time of a surrogate model and the total duration of the associated grid searches (Table IV - the first six columns), MLPs seem to be the best modeling method for our considered problems.

B. Ensemble-based Surrogates

In the second series of tests we checked whether using an ensemble strategy would improve the quality of the constructed surrogates. These ensemble modeling results are presented in the first two columns of Table III and they indicate that surrogates based on simple averaging ensembles of 10 base models are able to outperform single-model surrogates. In the case of ensembles as well, the model selection method based on thresholding ($MLP - ENS^{thr(5)}(10)$) seems to provide a very slight edge over simply selecting the 10 best performing cross-validation models.

For all the non-linear modeling methods considered in this study, model training time is positively correlated with the size of the training set. As such, the last tests that we performed were aimed at finding out if ensemble-based surrogate models trained on reduced training sets would also display a good and stable prediction accuracy.

Using the methods described in Section II-B we constructed training sets of 250 samples ($rand(250, 25)$ - with

TABLE II. INFORMATION REGARDING THE PREDICTION ACCURACY OF SINGLE-MODEL SURROGATES

Target	R^2 on test data				
	Linear	$MLP-SIN^{best}_{full(25)}$	$MLP-SIN^{thr(5)}_{full(25)}$	$SVR-SIN^{best}_{full(25)}$	$RBF-SIN^{best}_{full(25)}$
T1	0.8594	**0.9636**	0.9619	0.8917	0.8842
T2	0.7712	0.9390	**0.9531**	0.8886	0.7687
T3	0.7909	0.9434	0.9577	**0.9584**	0.8117
T4	0.9201	0.9943	0.9957	**0.9964**	0.9874
T5	0.9245	**0.9894**	0.9816	0.9831	0.9784
T6	0.8934	0.9616	**0.9644**	0.9637	0.9500
Average	0.8599	0.9652	**0.9690**	0.9469	0.8967
Rank	5	2	1	3	4

TABLE III. INFORMATION REGARDING THE PREDICTION ACCURACY OF ENSEMBLE-BASED SURROGATES

Target	R^2 on test data			
	$MLP-ENS^{best}_{full(25)}(10)$	$MLP-ENS^{thr(5)}_{full(25)}(10)$	$MLP-ENS^{thr(5)}_{rand(250,25)}(10)$	$MLP-ENS^{thr(5)}_{trim(250,25)}(10)$
T1	0.9586	**0.9637**	0.9525 ± 0.0054	0.9460 ± 0.0028
T2	**0.9556**	0.9549	0.9510 ± 0.0015	0.9517 ± 0.0050
T3	0.9573	**0.9596**	0.9502 ± 0.0108	0.9562 ± 0.0032
T4	**0.9961**	0.9960	0.9936 ± 0.0034	0.9920 ± 0.0035
T5	**0.9889**	0.9887	0.9865 ± 0.0020	0.9856 ± 0
T6	**0.9763**	0.9752	0.9739 ± 0.0010	0.9772 ± 0
Average	0.9721	**0.9730**	0.9676	0.9681
Rank	2	1	4	3

TABLE IV. INFORMATION REGARDING THE AVERAGE TRAINING TIMES OF THE SURROGATE MODELS AND THE TOTAL TIME REQUIRED BY THE BEST PARAMETER GRID SEARCHES

Target	Surrogate model training time [s]							
	$MLP-full(25)$		$SVR-full(25)$		$RBF-full(25)$		$MLP-trim(250,25)$	
	avg.	total	avg.	total	avg.	total	avg.	total
T1	88.22	70627	307.96	195248	112.59	62603	**12.35**	**10662**
T2	85.84	74081	291.99	185125	113.66	63539	**15.51**	**13447**
T3	73.93	63661	330.39	209140	114.46	63640	**12.81**	**11064**
T4	32.25	28313	120.82	77930	302.05	170960	**12.34**	**10875**
T5	45.96	55883	106.68	68810	212.85	120478	**12.46**	**15152**
T6	34.28	43298	114.70	74100	217.32	123007	**11.02**	**13399**
Average	60.08	55977.2	212.09	135058.8	178.82	100704.5	**12.74**	**12433.17**
Rank	2	2	4	4	3	3	1	1

a non-dominated sorting Pareto selection and $rand(25)$) and then we applied the $MLP-ENS^{thr(5)}(10)$ surrogate model construction strategy. Because both methods of constructing reduced training sets have (by design) a stochastic component, we repeated each modeling experiment five times and, in the last two columns of Table III, we present the obtained average and standard deviation values. The results indicate that, for the chosen parameters, there is no difference between applying a random and a Pareto-based trimming of the original training sets. However, the generational R^2 plots from Figure 2 show that the overall quality of the obtained surrogates is quite high (almost as good as $MLP-SIN^{thr(5)}_{full(25)}$ - the best performing single-model surrogate construction strategy) and that the predictions are also very stable (even when comparing with $MLP-ENS^{thr(5)}_{full(25)}(10)$ - the overall best performing surrogate model construction strategy).

The average training time of the base surrogate models and the total duration of the associated grid searches (Table IV - the last two columns) is more than four times smaller when using the random or Pareto-trimmed training sets than when training the MLPs on the full training sets. As Figure 3 shows, the distributions of MLP model training times when applying the grid searches on the trimmed training sets are much more compact as only a handful of the constructed models require a training time larger than 75 seconds. This aspect is very important when wanting to distribute the best parameter grid searches over a cluster or grid computing environment as it makes load balancing much easier.

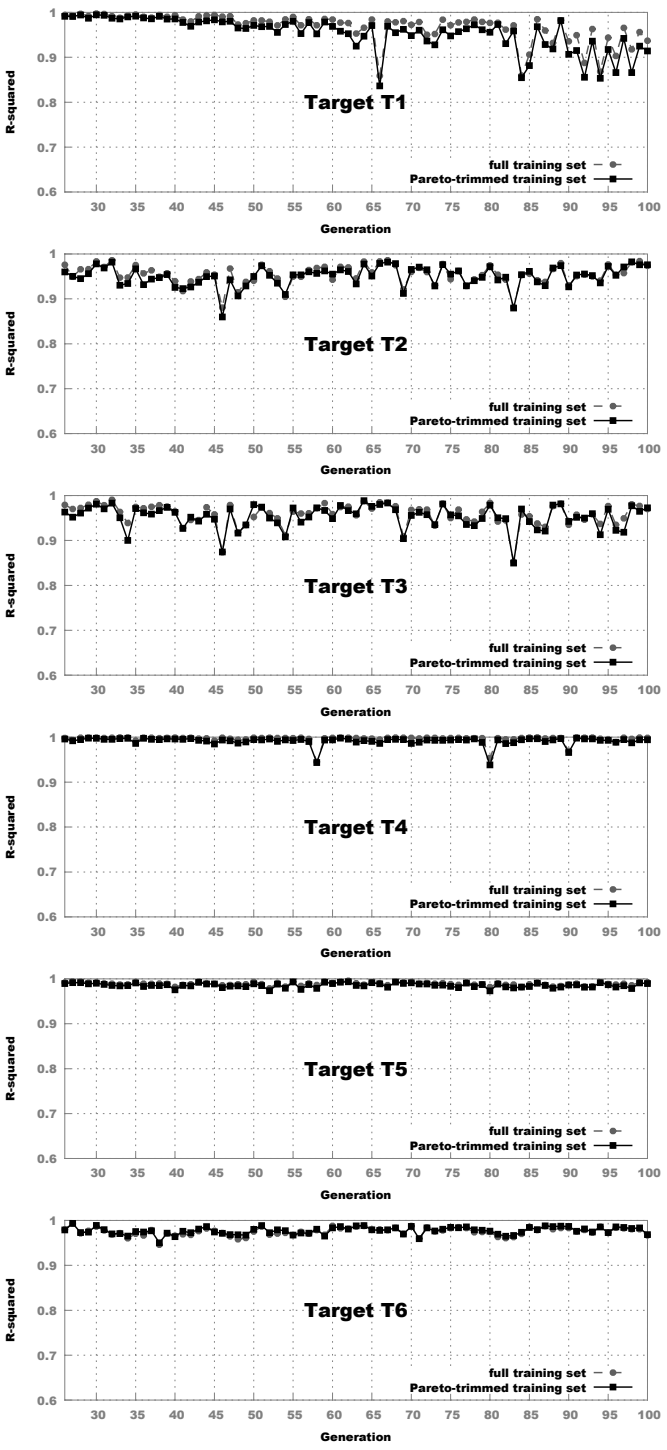

Fig. 2. Evolution of the generational coefficient of determination for $MLP-ENS^{thr(5)}_{full(25)}(10)$ and $MLP-ENS^{thr(5)}_{trim(250,25)}(10)$ (median accuracy model) for the six considered targets

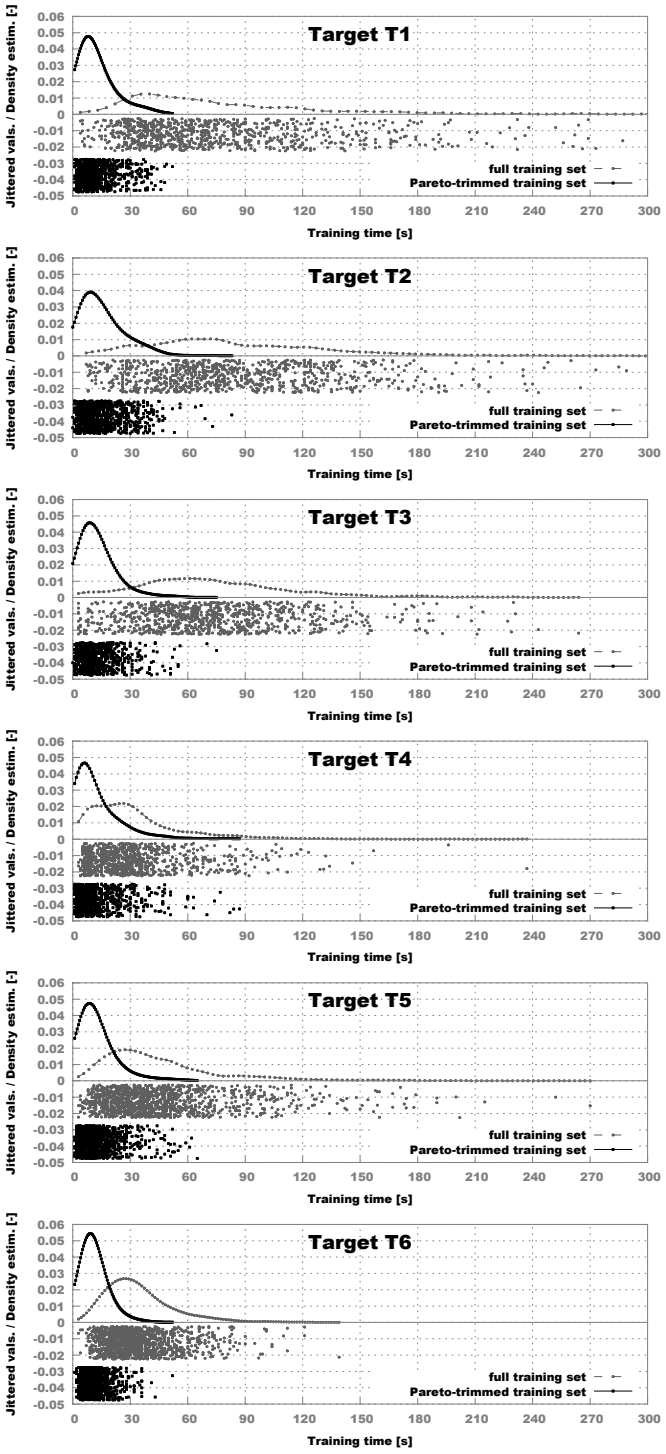

Fig. 3. Comparative kernel density estimations of the individual model training times observed when applying the MLP best parameter grid searches on the $full(25)$ and on the $trim(250,25)$ training sets for the six considered targets

V. Conclusions and Future Work

In this paper we described a fairly simple, but efficient, two stage approach for constructing on-the-fly surrogate models for multi-objective evolutionary algorithms used to optimize the design of electrical drives. In our case, surrogate modeling is needed because the original fitness function used by the evolutionary algorithms requires very time-intensive FE simulations.

As we propose to model each target (optimization objective or constraint) independently, the most important part of our surrogate training process lies in the construction of surrogates for the targets that display a non-linear dependency to the input

parameters. The results we obtained when trying to solve this task for three industrial data sets indicate that a strategy that uses ensembles of MLPs and Pareto-trimmed training sets (see $MLP-ENS^{thr(5)}_{trim(250,25)}(10)$) is able to deliver surrogate models that display the best accuracy vs. training time trade-off. When considering only the accuracy and stability over time of the surrogate predictions, a surrogate model construction strategy that uses MLPs trained using all the available samples (see $MLP-ENS^{thr(5)}_{full(25)}(10)$) is the overall best performer.

In the future we shall analyze how the two previously mentioned ensemble strategies work on other optimization scenarios, especially during on-line tests. The ability to create high-quality surrogates much faster (particularly when using a cluster computing environment) has the lateral effect of enabling us to execute the surrogate model construction stage several times during the optimization run. We plan to test if such an approach can deliver better and/or faster optimization results than the method described in [12] that performs only one surrogate construction stage.

Finally, we also plan to perform more tests in order to determine the qualitative difference of the resulting surrogate models obtained when using Pareto-trimmed (i.e., problem specific) and randomly-trimmed training sets constructed with other parameterizations (e.g., $rand(400,50)$ and $trim(400,50)$).

ACKNOWLEDGMENT

This work was conducted in the frame of the research program at the Austrian Center of Competence in Mechatronics (ACCM), a part of the COMET K2 program of the Austrian government. The work-related projects are kindly supported by the Austrian government, the Upper Austrian government and the Johannes Kepler University Linz. The authors thank all involved partners for their support. This publication reflects only the authors' views.

REFERENCES

[1] T. Johansson, M. Van Dessel, R. Belmans, and W. Geysen, "Technique for finding the optimum geometry of electrostatic micromotors," *IEEE Transactions on Industry Applications*, vol. 30, no. 4, pp. 912–919, jul/aug 1994.

[2] X. Jannot, J. Vannier, C. Marchand, M. Gabsi, J. Saint-Michel, and D. Sadarnac, "Multiphysic modeling of a high-speed interior permanent-magnet synchronous machine for a multiobjective optimal design," *IEEE Transactions on Energy Conversion*, vol. 26, no. 2, pp. 457 – 467, June 2011.

[3] Y. del Valle, G. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, and R. Harley, "Particle swarm optimization: Basic concepts, variants and applications in power systems," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 171–195, April 2008.

[4] L. V. Santana-Quintero, A. A. Montao, and C. A. C. Coello, "A review of techniques for handling expensive functions in evolutionary multi-objective optimization," in *Computational Intelligence in Expensive Optimization Problems*, ser. Adaptation, Learning, and Optimization, Y. Tenne, C.-K. Goh, L. M. Hiot, and Y. S. Ong, Eds. Springer, 2010, vol. 2, pp. 29–59.

[5] N. Queipo, R. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. Kevin Tucker, "Surrogate-based analysis and optimization," *Progress in Aerospace Sciences*, vol. 41, no. 1, pp. 1–28, 2005.

[6] Y. S. Ong, P. B. Nair, A. J. Keane, and K. W. Wong, "Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems," in *Knowledge Incorporation in Evolutionary Computation*. Springer, 2005, pp. 307–331.

[7] A. Forrester, A. Sobester, and A. Keane, *Engineering Design via Surrogate Modelling: A Practical Guide*. Wiley, 2008.

[8] Y. Tenne and C. Goh, Eds., *Computational Intelligence in Expensive Optimization Problems*, ser. Adaptation, Learning and Optimization. Springer, 2010, vol. 2.

[9] F. Neri, X. del Toro Garcia, G. L. Cascella, and N. Salvatore, "Surrogate assisted local search in PMSM drive design," *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, vol. 27, no. 3, pp. 573–592, 2008.

[10] F. Bittner and I. Hahn, "Kriging-assisted multi-objective particle swarm optimization of permanent magnet synchronous machine for hybrid and electric cars," in *IEEE International Electric Machines & Drives Conference (IEMDC 2013)*. IEEE, 2013, pp. 15–22.

[11] C. Coello, G. Lamont, and D. Van Veldhuisen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, ser. Genetic and Evolutionary Computation Series. Springer, 2007.

[12] A.-C. Zăvoianu, G. Bramerdorfer, E. Lughofer, S. Silber, W. Amrhein, and E. P. Klement, "Hybridization of multi-objective evolutionary algorithms and artificial neural networks for optimizing the performance of electrical drives," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 8, pp. 1781–1794, 2013.

[13] Y. Jin, M. Hüsken, M. Olhofer, and B. Sendhoff, "Neural networks for fitness approximation in evolutionary optimization," in *Knowledge Incorporation in Evolutionary Computation*, ser. Studies in Fuzziness and Soft Computing, Y. Jin, Ed. Springer, 2004, pp. 281–305.

[14] Y.-S. Hong, H. Lee, and M.-J. Tahk, "Acceleration of the convergence speed of evolutionary algorithms using multi-layer neural networks," *Engineering Optimization*, vol. 35, no. 1, pp. 91–102, 2003.

[15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[16] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization," in *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*. International Center for Numerical Methods in Engineering (CIMNE), 2002, pp. 95–100.

[17] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, pp. 115–148, 1995.

[18] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Computer Science and Informatics*, vol. 26, pp. 30–45, 1996.

[19] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software: an update," *SIGKDD Explorations*, vol. 11, no. 1, pp. 10–18, November 2009.

[20] P. J. Werbos, "Beyond regression: New tools for prediction and analysis in the behavioral sciences," Ph.D. dissertation, Harvard University, 1974.