Transient Modeling of Induction Machine Using Artificial Neural Network Surrogate Models

Mikko Tahkola[1], Victor Mukherjee[2], and Janne Keränen[1]

[1]VTT Technical Research Centre of Finland Ltd., 02044 Espoo, Finland
[2]Technology Center, ABB Motors and Generators, 00380 Helsinki, Finland

A transient model of an induction machine (IM) is developed in this work using an artificial neural network (ANN) surrogate model. The model is suitable to be used for direct-on-line IMs. The finite-element (FE)-based model of IM is used to generate the training, validation, and testing datasets. Different inputs and model configurations are investigated to find an optimal solution in developing the transient model. The proposed transient model is suitable to be used in digital twin services since it can estimate the current and torque accurately in real time based on only voltage and measured shaft speed.

Index Terms—**Artificial neural network (ANN), digital twin, induction machine (IM), real time, surrogate modeling.**

I. Introduction

Assessing performance of an induction machine (IM) over a startup period of time through its transient behavior is extremely important as most industrial pumps are still run by direct-on-line induction motors. The modeling of the steady-state and transient behaviors of the IM for real-time applications can be done with an analytical model and its accuracy can be improved by tuning different operational coefficients from finite-element (FE) computations. However, conventional equivalent circuit works only with the fundamental harmonic of the system and thus ignores all the effects of different spatial harmonics of the machine in the current and torque behavior. For IMs, the eddy current in the rotor bar and the saturation of the iron core make it further challenging to produce an accurate analytical model, especially at the transient state when the shaft is accelerating [1]. We demonstrate that an artificial neural network (ANN)-based surrogate model can learn the startup transient behavior of an IM from FE simulations and predict it quickly for real-time applications with almost similar accuracy as an FE model.

ANNs are a data-driven approach that does not necessarily require information about the model structure. Instead, a suitable ANN structure for a specific problem is found within hyperparameter optimization procedure, which is part of the surrogate modeling process [2]. ANNs were chosen for the study over deep and physics-informed neural networks as ANNs are often faster to develop than the former and easier to set up than the latter. The use of simpler regression models was not considered as ANNs offer more flexibility in modeling nonlinearities. Only a few published articles can be found in the literature on the application of ANNs for modeling the transient behavior of IM with FE to the best knowledge of the authors. These articles are mostly focused on condition monitoring [3], optimization [4], [5], and in the improvements of different analytical equation-based control strategies [6].

The goal of this work is to create an ANN-based IM model that accurately predicts the transient and steady-state current and torque. The application for this model is to monitor and optimize large industrial systems that comprise several IMs of different sizes and ratings in real time as a part of digital twin services. We study what inputs are required for an ANN-based model in different scenarios, by comparing the ANN performance with multiple input configurations. The input options include the electric current, the grid voltage, measured shaft speed, and first backward finite difference of the shaft speed. In addition, another objective of this work is to study whether current is essentially required as an input parameter to the ANN model to predict the transient torque behavior. It is worth mentioning that in case of direct online machines that are deployed in the service field, excluding current measurement makes the ANN model deployment for digital twin services easier and cheaper.

II. Induction Machine Modeling

A 45 kW double-cage induction motor is selected as a case study for this work, as shown in Fig. 1. Details for this machine can be found in [1]. The FE dataset for surrogate model development includes 2-D transient simulations from standstill condition to different power levels. The initial current and speed for the simulation are kept at zero, and next, the rotor is accelerated using a load profile to attain a specific steady speed, which generates the required power level. In this dataset, 11 power levels from 10 to 60 kW with 5 kW steps are selected as steady-state condition. Five of these power levels are used for training the ANNs, three for validating the ANNs, and the rest for testing them. The division of power levels to these three datasets is shown in Figs. 2 and 3, which shows the torque and current behavior of the machine at different power levels. The test cases include, in addition to a case with 30 kW power, both the minimum and the maximum power level to measure the extrapolation capability of the developed ANN

Manuscript received 4 February 2022; revised 6 April 2022; accepted 25 May 2022. Date of publication 3 June 2022; date of current version 26 August 2022. Corresponding author: M. Tahkola (e-mail: mikko.tahkola@vtt.fi).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TMAG.2022.3180176.

Digital Object Identifier 10.1109/TMAG.2022.3180176

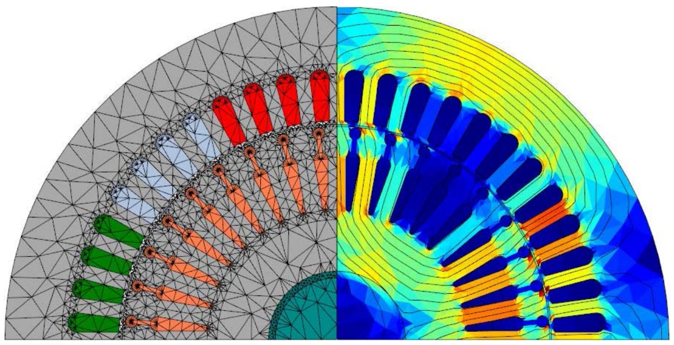
Fig. 1. Mesh and magnetic flux density of the case study machine at rated operation.

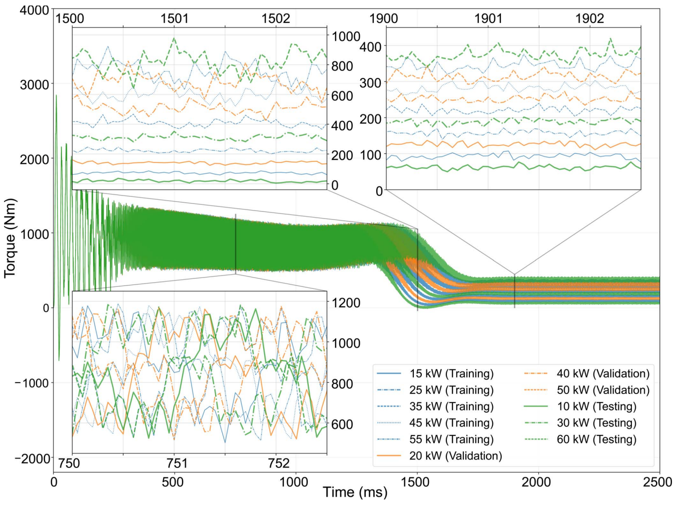

Fig. 2. Simulated torque at various target steady-state torque levels from standstill condition.

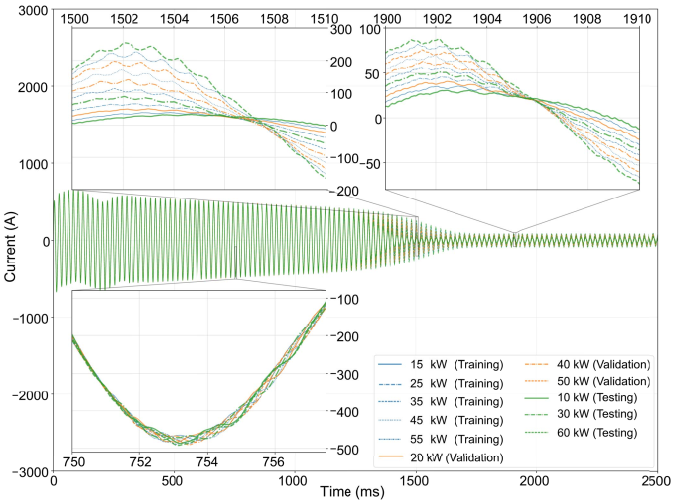

Fig. 3. Simulated current (phase A) at various target steady-state torque levels.

models. Each simulation consists of 80 000 timesteps, i.e., the total number of samples in the FE dataset is 880 000.

III. ANN Surrogate Model Development

Several input configurations are evaluated to find out the most optimal one for an ANN surrogate model to predict the torque and current accurately. The models are developed using

TABLE I
Input Configurations of the Trained ANN Surrogate Models, Showing From Which Timesteps Values of Variables Are Included in the Model Input

Abbrev.	I	U	ω	$\Delta\omega$
M1	k-1	k	k, k-1	-
M2	-	k	k, k-1	-
M3	-	k, k-1, ..., k-4	k, k-1, ..., k-4	k, k-1, ..., k-4

gated recurrent units (GRUs) [7], using the Tensorflow Python library [8]. The units in a GRU network have recurrent connections, i.e., they take their own output at previous timestep $k-1$ as an input at timestep k, in addition to actual input values. Computations related to the training and the evaluation of the models presented in this work were done using an Intel i9-9900K central processing unit.

The model input options include three-phase voltage U (phases A, B, and C), shaft speed ω, first backward finite difference of shaft speed $\Delta\omega$ ($\Delta\omega_k = \omega_k - \omega_{k-1}$), and three-phase current I (phases A, B, and C) of previous timestep. The model outputs include three-phase current I and torque T. The studied input configurations are presented in Table I. The first input configuration, referred to as $M1$, includes three-phase voltage at timestep k, shaft speed ω at timesteps k and $k-1$, and current I at previous timestep $k-1$. Such input configuration allows to use the model in only circumstances where current measurement is available, which is rarely the case. The second input configuration, referred to as $M2$, has the same configuration as the first one, excluding the currents. With $M2$, it is evaluated how much the lack of current information affects the prediction error of the surrogate model in this case. The third input configuration, referred to as $M3$, includes the first backward finite difference of shaft speed $\Delta\omega$ in addition to the voltages U and shaft speed ω, each at the current and four previous timesteps, i.e., at $k, \ldots, k-4$. Our hypothesis is that adding $\Delta\omega$ improves the predictive performance of the ANN since T is linearly dependent on the time derivative of the ω. With the $M3$ configuration, we experimented with various numbers of timesteps in the input. Five timesteps showed the best performance in terms of prediction error and the results obtained with this configuration are shown in this article.

Both the input and the output values are each independently standardized by removing the mean and scaling to unit variance. To prevent information leakage from the training dataset to validation and testing datasets, the mean and standard deviation of each variable are computed based on samples of the training dataset only, and the same values are used to standardize the samples in the validation and testing datasets.

In this work, Hyperopt Python library [9] is used to optimize hyperparameters of the ANN (number of layers and units) and learning algorithm (learning rate and batch size). The optimization algorithm was given an option to choose between Adam and Adamax learning algorithms to be used in training the model. Hyperopt is a sequential model-based optimization algorithm, which fits an internal surrogate model to map the relationship between hyperparameters and a selected error quantity, with the objective to find hyperparameters that

TABLE II
HYPERPARAMETER SEARCH SPACE AND OPTIMIZED HYPERPARAMETERS FOR EACH INPUT CONFIGURATION

		Optimized values		
Hyperparameter	Values	M1	M2	M3
Number of layers	1,2,3,4	4	2	2
Number of units	32,64,128	256	64	64
Optimizer	Adam, Adamax	Adam	Adam	Adamax
Batch size	256, 512, 1024, 2048	1 024	2 048	1 024
Learning rate	1e^{-4}–5e^{-2}	5e^{-3}	1.02e^{-2}	2.04e^{-2}

TABLE III
TORQUE AND CURRENT PREDICTION ERROR OF THE ANN SURROGATES WITH THREE INPUT CONFIGURATIONS WITH THE TEST DATASET

Output	Configuration	RMSE	NRMSE	MAE
Torque (Nm)	M1	26.3	0.7 %	15.6
	M2	107.4	3.0 %	67.2
	M3	23.7	0.7 %	15.8
Current (A)	M1	1.3	0.1%	0.9
	M2	8.9	0.6%	6.3
	M3	3.1	0.2%	2.3

minimizes this quantity [9]. Here, the optimization algorithm minimizes the mean squared error (MSE) computed on the validation dataset, but also the root-mean-squared error (RMSE), the normalized (N)RMSE, and the mean absolute error (MAE) are recorded. The NRMSE is computed by dividing the RMSE by the difference of maximum and minimum values of a case. The optimization algorithm was set to first explore 25 pseudorandomly selected hyperparameter combinations to initialize the internal surrogate model, after which the algorithm uses its internal surrogate model to select the next 35 hyperparameter combinations. After each combination, the internal surrogate model is updated before selecting the next hyperparameters to try. The hyperparameters, their allowed values for optimization, and the optimized values for each input configuration are shown in Table II.

The time required by the hyperparameter optimization was 50, 4, and 10 h with the configurations $M1$, $M2$, and $M3$, respectively. The differences stem from the optimization converging toward different values, e.g., neural network and batch sizes, as shown in Table II. These hyperparameters affect the time required to train a model. The computational efficiency could be enhanced by parallel computation of the random iterations in the hyperparameter optimization.

Training of a model starts by defining the neural network structure and learning related hyperparameters. The number of training iterations was set to 10 000. The learning algorithm was set to utilize 30% of the samples in the training dataset to monitor how the prediction error evolves during training. An early stopping mechanism was used to stop the training once the monitored prediction error had not improved from the currently lowest error in the previous 30 training iterations.

After the hyperparameter optimization, the models were ranked by their prediction error (RMSE) computed on the validation dataset and the model with the lowest RMSE was selected as the final model.

IV. PERFORMANCE OF ANN SURROGATE MODELS

The results of ANN model development following the workflow presented in Section III are presented in this section. The same workflow was used to develop a model with each input configuration. The results shown in this section are computed with a test dataset that was used solely for testing the selected final model with each input configuration.

The performance of the three different ANN models in predicting the torque and the current transient behavior of the test cases is shown in Table III. The ANN that is trained with current from the previous timestep in input ($M1$) has achieved an acceptable RMSE of 26 N $\cdot$ m on the test case, while without the knowledge of previous timestep current value ($M2$), the RMSE is 107.4 N $\cdot$ m. The ANN with input configuration $M1$ has also predicted three-phase current at current timestep with an RMSE of 1.3 A, while the ANN-M2 has an RMSE of 8.9 A. This shows that the ANNs are unable to capture the torque (T_k) and current (I_k) behavior based on voltage and raw shaft speed values only. However, when the first backward finite difference of the shaft speed is added to the model input ($M3$), the torque and current RMSEs are decreased to 23.7 N $\cdot$ m and 3.1 A, respectively. The MAE for the ANN-M1 and ANN-M3 is almost equal (15.58 and 15.68 N $\cdot$ m, respectively). This result demonstrates the importance of feature engineering in data-driven model development.

Torque predictions with the three models on an extrapolation test case with 60 kW power level are shown in Fig. 4. During the acceleration, i.e., the first 750 ms after the machine has started, the predictions of the two best models follow the FE simulation result accurately. At 1500 ms, the machine reaches the steady state, and there, the $M3$ model with $\Delta\omega$ in the input can predict the torque harmonics better than the model with current in the input. Similar behavior can be seen at the steady-state operation, where the ANN-M3 predicts the torque more accurately than the model with current in the input ($M1$).

Current predictions of the ANN-M3 similarly follow the FE simulation result accurately, as shown in Fig. 5. Naturally, the model with previous timestep current I_{k-1} can predict the current I_k more accurately than the models without the current in the input. Still, the current prediction accuracy of model with difference of shaft speed in the input approximates the current relatively well.

The torque results support our hypothesis that $\Delta\omega$ in the input helps to predict the torque. In addition, the current is also co-related to torque by magnetomotive force. Hence, the co-relation of the shaft speed difference with torque also enables it to correlate with current. Therefore, the $M3$ model, even without prior knowledge of current from previous timesteps, is still able to predict the current accurately. This capability of the data-driven ANN model shows that it can overcome the complexity of modeling the eddy current effect of the rotor bar of IM.

The ANN model with the previous timestep current in input ($M1$) has predicted one sample in 34 μs on average, whereas the corresponding times for $M2$ and $M3$ were 12 and 25 μs, respectively. These correspond to approximately 30 000, 84 000, and 40 000 samples per second. Although $M2$ was the fastest with the least parameters, $M3$ is preferred due to its

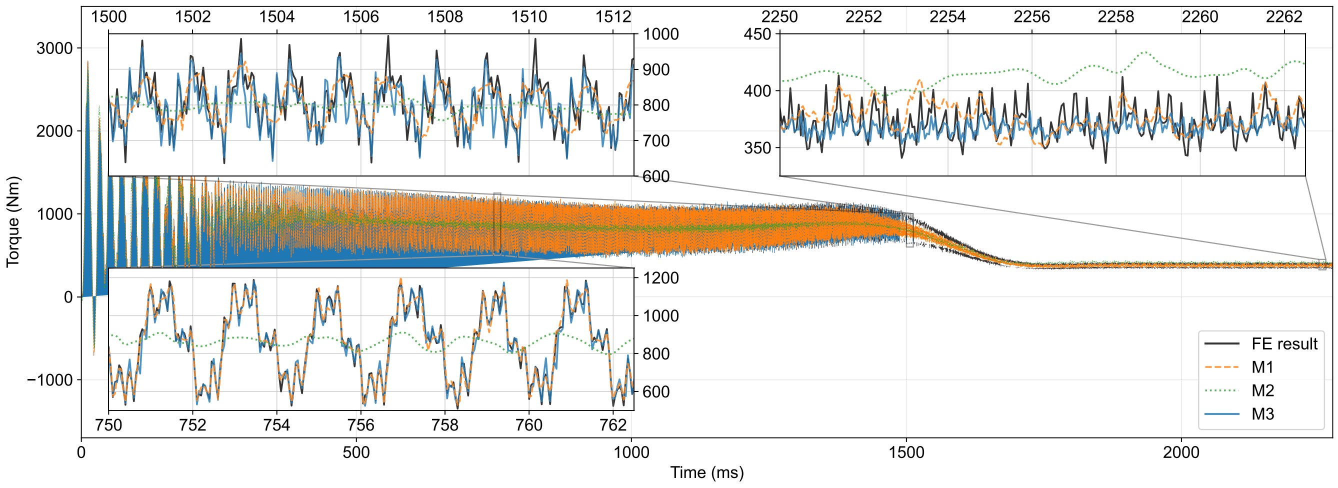

Fig. 4. Torque predictions of ANNs with different input configurations. Test case with a power of 60 kW.

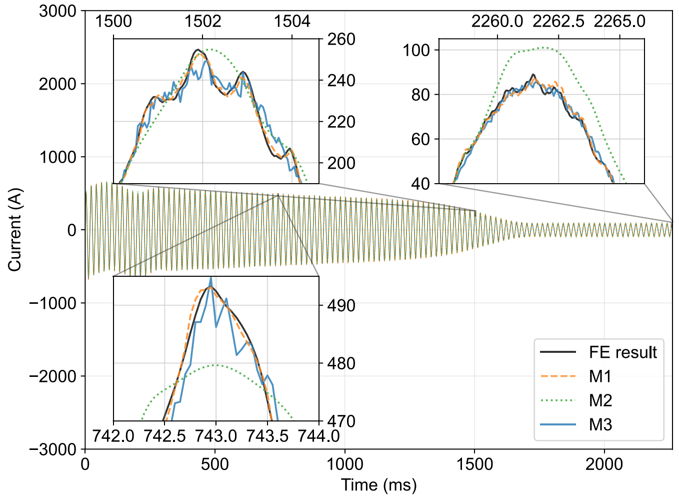

Fig. 5. Current (phase A) predictions of ANNs with different input configurations. Test case with a power of 60 kW.

better accuracy. The computation times were measured in a Python environment.

The required time to compute one timestep is 12 ms with the FE simulation used in this work and 8 μs with the analytical model as referred in [1]. Therefore, the computational time is reduced by a factor of 500 with the ANN ($M3$) when compared to the FE simulation. This shows that an ANN surrogate model can achieve an accuracy close to an FE model and a speed that is comparable to that of an analytical model.

V. Conclusion

The results demonstrate that an ANN can accurately and in real time predict the torque and current transient behavior of an IM. An ANN with voltage, shaft speed, and previous timestep current can predict torque behavior accurately, but the current measurement is rarely available. In this study, we have shown that by precomputing the first backward finite difference of the shaft speed and using it as an input to the ANN together with the voltages and the shaft speed, the ANN is capable of predicting the torque and current transient behavior without knowledge of the currents. In addition, the developed model was shown to extrapolate to both lower and higher power level than it was trained on. The next step is to validate the results using experimental data to either train a new model from scratch or adapt the developed model using transfer learning. In future, this kind of model can be further extended to optimize the operation of several industrial motors in real time. Also, different loss computations and fault analysis models can be built on top of the proposed ANN model.

Acknowledgment

This work was supported in part by the Arrowhead Tools project, funded by the European Commission through the European H2020 Research and Innovation Program, in part by the Electronic Components and Systems for European Leadership (ECSEL) Joint Undertaking, and in part by the National Funding Authorities from the 18 countries under Grant 826452.

References

[1] V. Mukherjee, T. Martinovski, A. Szucs, J. Westerlund, and A. Belahcen, “Improved analytical model of induction machine for digital twin application,” in *Proc. Int. Conf. Electr. Mach. (ICEM)*, Aug. 2020, pp. 183–189.

[2] M. Tahkola, J. Keranen, D. Sedov, M. F. Far, and J. Kortelainen, “Surrogate modeling of electrical machine torque using artificial neural networks,” *IEEE Access*, vol. 8, pp. 220027–220045, 2020.

[3] K. Kudelina, T. Vaimann, B. Asad, A. Rassõlkin, A. Kallaste, and G. Demidova, “Trends and challenges in intelligent condition monitoring of electrical machines using machine learning,” *Appl. Sci.*, vol. 11, no. 6, p. 2761, Mar. 2021.

[4] S. Barmada, N. Fontana, L. Sani, D. Thomopulos, and M. Tucci, “Deep learning and reduced models for fast optimization in electromagnetics,” *IEEE Trans. Magn.*, vol. 56, no. 3, pp. 1–4, Mar. 2020.

[5] I. Ibrahim, R. Silva, M. H. Mohammadi, V. Ghorbanian, and D. A. Lowther, “Surrogate models for design and optimization of inverter-fed synchronous motor drives,” *IEEE Trans. Magn.*, vol. 57, no. 6, pp. 1–5, Jun. 2021.

[6] X. Fu and S. Li, “A novel neural network vector control technique for induction motor drive,” *IEEE Trans. Energy Convers.*, vol. 30, no. 4, pp. 1428–1437, Dec. 2015.

[7] K. Cho *et al.*, “Learning phrase representations using RNN encoder–decoder for statistical machine translation,” 2014, *arXiv:1406.1078*.

[8] M. Abadi *et al.*, “TensorFlow: Large-scale machine learning on heterogeneous systems,” in *Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, vol. 2016, pp. 265–283.

[9] J. Bergstra, D. Yamins, and D. D. Cox, “Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 28, 2013, pp. 115–123.