Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

A "DATA DIGITAL TWIN" DEVELOPMENT TO ENHANCE THERMAL ESTIMATION IN ELECTRIC TRACTION MOTORS

David Blum[1], Fauzan Dhongre[1], Alekh Kurup[1], Jaehoon Jeong[1], Anandakumar Subbiah[1], Vignesh Krishnakumar[1], Anuradha Kodali[1], Ram Surada[1], Vivek Sundaram[1], and Dhyana Ramamurthy[1]

[1]Lucid Motors, Newark, CA, USA

ABSTRACT

Lucid Motors has begun a proof of concept which aims to formulate a motor thermal estimator using a modern time series transformer and integrate it into our current physics-based model, creating a "data digital twin" of the motor thermal system. Outputs include component-level heat and power loss estimates over time, which are important for tuning motor performance. Our goal with the data digital twin is to improve latency and increase the time step frequency for thermal estimation, which are currently limited by computational load and in turn limit the optimization and accuracy of control algorithms designed to react to the temperature gradients within the drive unit.

1 TECHNICAL DESCRIPTION

Electric motors are used in various applications like electric cars, planes, and ships. They convert electrical energy into mechanical energy, but some of this energy is lost as heat. Monitoring or predicting motor temperatures is crucial to prevent damage. While sensors can measure these temperatures, they might get damaged due to high heat. Therefore, thermal and fluid dynamics models are used to predict temperatures, though their complexity can limit simulation speed. For benchmarking, we have used one such physics-based model as a black box to provide component temperatures within the motor.

To fit a data model, we formulate an AI-based iTransformer. Measurement data from the Lucid motor dynamometer laboratory are assembled into a multi-variate GluonTS data object which is used to build a data loader and for feature generation and other helpful operations for training transformer models such as slicing, masking, and sampling. Variables for forecasting include oil temperatures, motor winding temperatures, rotor temperatures, and stator temperatures, with additional telemetry signals such as motor torque, pump speed feedback, and stator currents. Integration is performed in Mathworks Simulink.

The training process involves resampling the data to 1 kHz to match the controller's operating frequency, slicing the data to 0.4 second intervals, adding additional time series features, and finally training a multivariate iTransformer model in PyTorch.

2 MODEL SELECTION

The iTransformer is a modern time series forecasting neural network architecture that works by using many of the same generative AI concepts as the transformers that power large language models, like token embedding and self-attention. It is capable of capturing multivariate correlations and learning nonlinear representations across variates, demonstrating state-of-the-art performance in a variety of settings, especially in high-dimensional and long-lookback scenarios. Some settings where iTransformer has seen recent publication include battery life forecasting, traffic speed and motion prediction in autonomous driving, ocean surface temperature forecasting, disk scheduling in high performance I/O, short term electric load forecasting, and many others.

3 TRAINING, EVALUATION, INTEGRATION, AND TESTING

A guided hyperparameter search yielded a hyperparameter set that converged and trained well over 100 epochs. At the time of this submission, using 4 distinct training sets and an iterative fine-tuning process, and setting aside 1% of each training set for evaluation, we have achieved parameter-specific mean absolute scaled error values on the evaluation sets ranging from 4.53 to 0.0053, representing an accuracy that varies between ½ order of magnitude worse to 2 ½ orders of magnitude better than a random walk. We expect to see further improvement in model performance as we continue to incorporate additional training datasets, representing a wider variety of real-world conditions, and as we train and tune more exhaustively on those datasets.

Prior to acceptance, our digital twin models are correlated to data acquired from the vehicle under multiple operating conditions involving varying driving maneuvers, ambient temperatures, and vehicle loads, to an error that is acceptable for the specific target application. We will perform validation on the iTransformer model under various operating conditions using the physics-based digital twin, as well as evaluate compute load on the target microcontroller, by running them in parallel on the same sets of inputs via test harness.

4 CONCLUSION

The development of an AI-based motor thermal estimator model represents a significant advancement in the field of EV thermal management. By combining data-driven techniques with traditional physics-based models, this approach aims to provide more accurate and reliable thermal estimates at reduced latency, ultimately allowing for safer and more durable operation of electric traction motors.

5 CONTACT INFORMATION

David Blum (*lead author*)
Lucid Motors
E-Mail: davidblum@lucidmotors.com

Fauzan Dhongre
Lucid Motors
E-Mail: fauzandhongre@lucidmotors.com

Alekh Kurup
Lucid Motors
E-Mail: alekhkurup@lucidmotors.com

Jaehoon Jeong
Lucid Motors
E-Mail: jaehoonjeong@lucidmotors.com

Anandakumar Subbiah
Lucid Motors
E-Mail: anandakumarsubbiah@lucidmotors.com

Vignesh Krishnakumar
Lucid Motors
E-Mail: vigneshkrishnakumar@lucidmotors.com

Anuradha Kodali
Lucid Motors
E-Mail: anuradhakodali@lucidmotors.com

Ram Surada
Lucid Motors
E-Mail: rambabusurada@lucidmotors.com

Vivek Sundaram
Lucid Motors
E-Mail: viveksundaram@lucidmotors.com

Dhyana Ramamurthy
Lucid Motors
E-Mail: dhyanaramamurthy@lucidmotors.com

REFERENCES

Yong Liu et al. 2024. iTransformer: Inverted Transformers are Effective for Time Series Forecasting. ICLR 2024. https://arxiv.org/pdf/2310.06625.

Alexander Alexandrov et al. 2020. GluonTS: Probabilistic and Neural Time Series Modeling in Python. JMLR 21(116):1-6.