Prediction of Remaining Useful Life and Cell Temperature for Li-ion Batteries using TinyML

Yuqin Weng
Dept. of Elec. and Comp. Engr.
Marquette University
Milwaukee, WI, USA
yuqin.weng@marquette.edu

Wenkai Guan
Div. of Science and Math.
University of Minnesota, Morris
Morris, MN, USA
guan0210@morris.umn.edu

Cristinel Ababei
Dept. of Elec. and Comp. Engr.
Marquette University
Milwaukee, WI, USA
cristinel.ababei@marquette.edu

***Abstract*—In this paper, we develop new tiny machine learning (tinyML) temporal convolutional network (TCN) models for prediction of remaining useful life (RUL) and of cell temperature for lithium-ion batteries. The proposed models are developed, trained, optimized and verified in Python using TensorFlow. Extensive simulation experiments, using datasets from the Battery Archive website and from Sandia National Lab (SNL), show that the proposed models provide better results compared to previous models. Furthermore, the proposed models are converted to TensorFlow lite for microcontroller models, which are deployed on IoT hardware devices, specifically the popular Arduino Nano 33 BLE Sense board. We conduct hardware experiments that show that the tinyML models are very efficient and provide satisfactory prediction accuracy. Therefore, the proposed optimized tinyML models could be easily deployed in real practical scenarios, such as electric vehicles (EVs), to continuously monitor in real-time the health and temperature of batteries.**

***Index Terms*—battery pack, electric vehicle, remaining useful life, thermal management, deep neural network, tinyML**

I. Introduction

Recent advancements in artificial intelligence (AI), machine learning (ML), and internet of things (IoT) motivated researchers to apply these technologies to the design and optimization of a lot of electronic devices and systems, including battery management systems in electric vehicles (EVs), portable power drills, defibrillators, and portable x-ray machines [1]. In the context of EVs, an important problem is the health management of battery packs, and many solutions to this problem involve or require prediction of attributes like the remaining useful life (RUL) and temperature of battery cells inside lithium-ion battery packs [2], [3]. In this paper, we present AI/ML models to predict both RUL and cell temperature of lithium-ion batteries and deploy them as tinyML models on IoT microcontroller devices.

II. Related Work

Several previous studies investigated the problem of prediction of RUL. RUL prediction techniques can be generally categorized into three classes: model-based [4], data-driven [5], [6], and hybrid approaches [7], [7], [8]. A few recent studies investigated quantized ML models for state of health (SoH) and state of charge (SoC) estimation [9], [10], but, not for RUL. Thermal management also plays an important role in battery management systems. Without proper temperature prediction and thermal management, batteries can overheat, increasing internal resistance and leading to side chemical reactions. Consequently, the overall cycle life of the battery can be shortened. In the literature, one can also find several AI approaches for battery cell temperature prediction, including supervised, unsupervised, and reinforcement learning [11].

Previous work that focused on developing models for RUL and/or battery cell temperature prediction without necessarily verifying them on IoT microcontroller devices. Instead, they stopped at simulation conducted on computers, where one can afford models that are very large but, with better performance. However, in many practical situations [12], we actually want such models to be implemented on IoT devices, which are typically built using microcontrollers, which are resource constrained (i.e., memory capacities in the order of 250KB only). Therefore, in this paper, we develop for the first time several deep neural network (DNN) models for prediction of RUL and of battery cell temperature and optimize them specifically for IoT devices using tinyML technologies.

III. Model Development and Testing Workflow

Before describing the investigated models, in this section, we first present the workflow that we follow in developing, optimizing, and and then deploying ML models for RUL and temperature cell prediction. This discussion will present a better overall context for the proposed work. In studying all the models, we follow the methodology described by the pipeline from Fig. 1.

First, as indicated in the top-right quadrant of the figure, datasets are obtained from Sandia National Lab, which makes available a variety of datasets collected from experiments using cycling tests for three commercial cells [13]. We selected two types of battery cells: Nickel Manganese Cobalt (NMC) and Nickel Cobalt Aluminum (NCA), because the cycling experiments used different values for temperature range, depth of discharge (DoD), and discharging current [14]. These datasets are split into train, validation, and test portions. The train and validation portions are used for model development. Model parameter tuning is done via empirical grid search methods. Once the models are developed and trained, they are evaluated on the test portion of the datasets - as shown in the bottom-right quadrant of the figure. Note that this step is done via

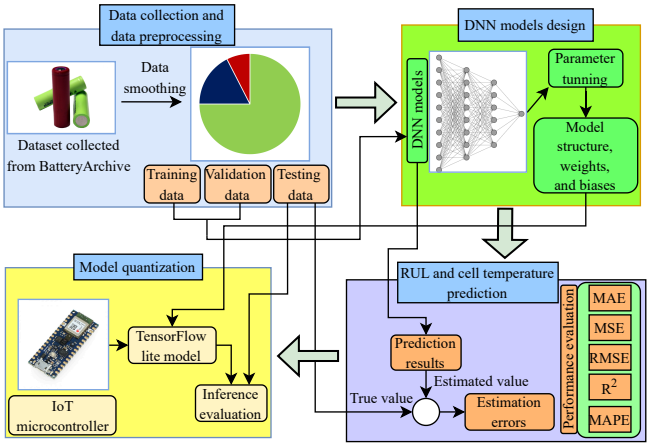


Fig. 1: Overall workflow for ML model development, optimization and testing on IoT device.

simulations in Python conducted on computers or servers in the cloud because the models at this stage are still TensorFlow models that have not been optimized (i.e., quantized) yet for deployment to microcontrollers.

Finally, the best found model is deployed on an IoT microcontroller. In this work, we use the Arduino Nano BLE 33 Sense, which has a Nordic nRF52840 (ARM Cortex-M4) processor running at 64MHz, with 1MB flash and 256KB of RAM [15]. The deployment on such microcontroller, with limited resources, is possible by using tiny machine learning technologies. More specifically, we use Google's TensorFlow Lite (TFL) and TensorFlow Lite for Microcontrollers (TFLM) [16] to convert the developed model into an optimized format that can be easily integrated into C/C++ applications run on the Arduino board. The primary optimization technique is quantization (to 8-bit integer precision), which helps reduce significantly the size of the model. The advantage of the post-training quantization is that the model will be smaller in storage size and memory usage and hence faster in terms of inference time. However, the trade-off is performance degradation.

IV. Proposed Temporal Convolution Network

We investigate three DNN models: 1-dimensional convolution neural network (1D CNN), combination of 1D CNN and gated recurrent unit (GRU) (1D CNN+GRU), and novel temporal convolutional network (TCN). In this section, we present a discussion of the proposed temporal convolution network (TCN), which has never been used for RUL and cell temperature prediction. A simplified model architecture is shown in Fig. 2. The TCN block is an adapted version of the one introduced in [17] as WaveNet for the purpose of generating raw audio waveforms. It is constructed with so-called dilated causal convolution networks, which are a special type of convolution in which the filter is applied upon an expended receptive field by skipping input data with a certain step. It has been shown that TCN blocks can be very effective in dealing with long time series data input sequences [18]. By stacking multiple dilated causal convolution layers together, a

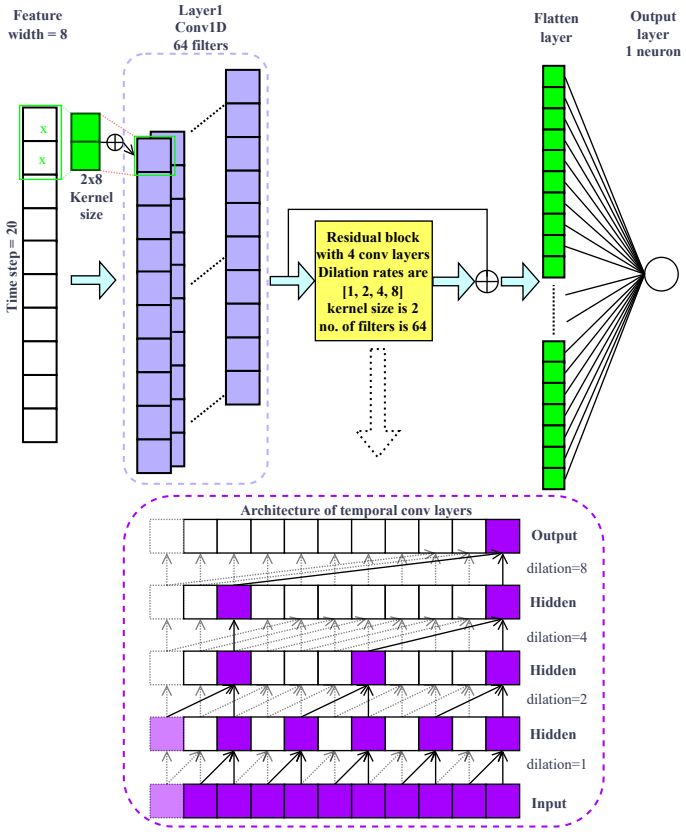


Fig. 2: Illustration of the TCN model developed in this paper to predict URL and cell temperature

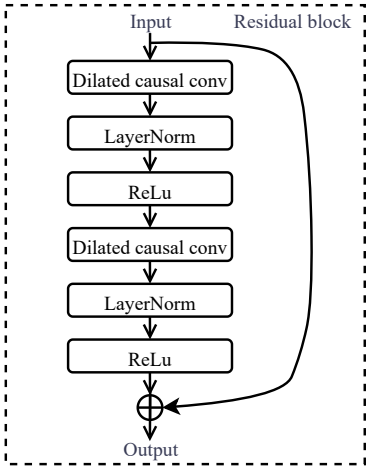


Fig. 3: Simplified diagram of the residual block.

deep neural network can be formed. Usually, associated with the TCN block is a residual block, which helps address the problem of gradient vanishing or exploding. Fig. 3 illustrates the residual block. The input to the residual block goes through a combination of dilated casual convolutions, layer normalizations (LayerNorm), and rectified linear unit (ReLu) activation functions.

As illustrated in Fig. 2, the input in the proposed TCN model has three dimensions: the number of samples (batch size), the time steps (20), and the feature width (8). The actual 8 features include: test-time (s), current (A), voltage

TABLE I: Listing of final model architecture parameters.

Model	Model output	Layer 1 kernels (kernel size)	Layer 2 kernels (kernel size)	GRU units
1D CNN	RUL	32 (4x8)	64 (2x8)	-
	Cell Temp.	8 (2x8)	16 (4x8)	-
1D CNN+GRU	RUL	32 (4x8)	-	64
	Cell Temp.	64 (4x8)	-	16
TCN	RUL	16 (4x8)	-	-
	Cell Temp.	32 (2x8)	-	-

TABLE II: RUL prediction errors (NMC dataset).

Models	Time steps	MAE	MSE	RMSE	R2
1D CNN	1	5.33 ± 0.42	48.41 ± 7.47	6.94 ± 0.54	1 ± 0
	3	5.62 ± 0.63	57.88 ± 15.55	7.53 ± 1.11	1 ± 0
	6	**5.18 ± 0.83**	**46.31 ± 16.32**	**6.69 ± 1.25**	**1 ± 0**
	12	5.74 ± 1.25	63.94 ± 27.08	7.82 ± 1.66	1 ± 0
	18	5.83 ± 0.77	59.26 ± 17.86	7.59 ± 1.27	1 ± 0
1D CNN+GRU	1	1.40 ± 0.14	3.95 ± 0.98	1.97 ± 0.25	1 ± 0
	3	1.37 ± 0.12	4.15 ± 1.54	2.00 ± 0.36	1 ± 0
	6	1.49 ± 0.09	4.46 ± 0.78	2.10 ± 0.18	1 ± 0
	12	1.57 ± 0.15	5.47 ± 0.92	2.33 ± 0.20	1 ± 0
	18	**1.23 ± 0.10**	**3.58 ± 1.32**	**1.86 ± 0.34**	**1 ± 0**
TCN	1	0.93 ± 0.27	1.61 ± 1.14	1.2 ± 0.4	1 ± 0
	3	0.90 ± 0.22	1.37 ± 0.67	1.14 ± 0.27	1 ± 0
	6	0.88 ± 0.2	1.34 ± 0.61	1.13 ± 0.26	1 ± 0
	12	**0.74 ± 0.04**	**0.87 ± 0.09**	**0.93 ± 0.05**	**1 ± 0**
	18	0.82 ± 0.23	1.17 ± 0.76	1.04 ± 0.30	1 ± 0

(V), charge and discharge capacity (Ah), charge and discharge energy (Wh), and environment temperature (C). The first layer is a 1D CNN layer with 64 filters or kernels of size 2x8. The output of this layer feeds into the TCN block, which internally has four instances of the residual block; and each such instance uses different dilation values, namely 1,2,4,8 (found based on our empirical investigations). Finally, the TCN block is followed by a flatten layer and the output layer with only one neuron because this proposed TCN network model is used for regression or prediction of either RUL or cell temperature. Two separate such models will be trained, one for each of the two target variables, RUL and cell temperature. Table I lists the parameters used for the two TCN models. This table also lists the parameters used for the other two investigated models (1D CNN and 1D CNN+GRU), which represent the state-of-the-art that we use for comparison purposes in this paper.

V. Performance Evaluation

In this section, we evaluate the performance of all three models. In the first phase of our experiments, we first conduct simulations using the NMC dataset. At this stage, all simulations are conducted in Python and all models are developed using TensorFlow. To help improve the performance of the models, we used a data smoothing technique described in [19], which essentially replaces feature values in a certain number of time steps with the mean of those values. In order to identify the best smoothing, we investigated five different numbers of steps for the smoothing technique. The performance of all the three models investigated in this paper to predict RUL and cell temperature - developed and tuned using five different versions of the datasets (corresponding to the five different variations of the smoothing technique) - are reported in Tables II and III. For robustness and reliability of the reported results, the same experiment is conducted five different times and the mean and

TABLE III: Cell temperature prediction errors when using the NMC dataset.

Models	Time steps	MAE	MSE	RMSE	R2
1D-CNN	1	0.38 ± 0.01	0.46 ± 0.05	0.68 ± 0.04	0.87 ± 0.01
	3	0.37 ± 0.03	0.41 ± 0.08	0.64 ± 0.06	0.89 ± 0.02
	6	0.36 ± 0.02	0.39 ± 0.04	0.63 ± 0.04	0.89 ± 0.02
	12	0.31 ± 0.02	0.30 ± 0.05	0.55 ± 0.05	0.91 ± 0.02
	18	**0.30 ± 0.01**	**0.30 ± 0.05**	**0.55 ± 0.05**	**0.88 ± 0.02**
1D CNN+GRU	1	0.50 ± 0.44	1.03 ± 1.38	0.86 ± 0.55	0.72 ± 0.37
	3	**0.28 ± 0.01**	**0.33 ± 0.03**	**0.58 ± 0.03**	**0.91 ± 0.01**
	6	0.48 ± 0.43	0.95 ± 1.32	0.81 ± 0.55	0.73 ± 0.37
	12	0.45 ± 0.41	0.81 ± 1.15	0.74 ± 0.51	0.74 ± 0.38
	18	0.44 ± 0.40	0.72 ± 0.98	0.71 ± 0.47	0.73 ± 0.37
TCN	1	0.25 ± 0.02	0.31 ± 0.07	0.55 ± 0.06	0.92 ± 0.02
	3	0.25 ± 0.01	0.27 ± 0.04	0.52 ± 0.04	0.92 ± 0.01
	6	0.23 ± 0.02	0.24 ± 0.05	0.49 ± 0.04	0.93 ± 0.01
	12	0.22 ± 0.01	0.23 ± 0.02	0.48 ± 0.02	0.93 ± 0.01
	18	**0.21 ± 0.02**	**0.2 ± 0.03**	**0.45 ± 0.04**	**0.92 ± 0.01**

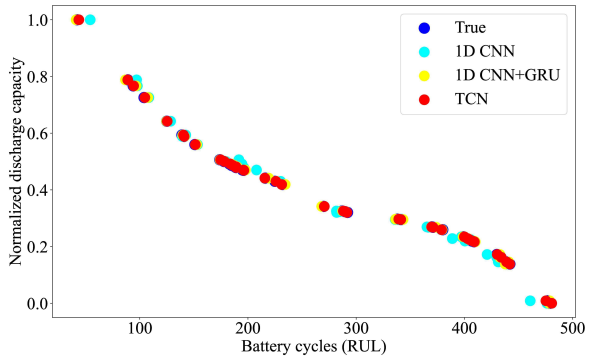
the standard deviation (std) values of all five experiments are reported in these tables. The best result for each model is highlighted in bold font. We observe that the proposed TCN model performs better than the other two models.

In addition to the results reported in the table, we present the plots from Fig. 4. Fig. 4.a plots the relationship between the normalized discharge capacity and RUL. The discharge capacity in this plot is captured for time steps at the end of each batter cycle. Therefore, the number of data points plotted on this figure does not equal the number of data points in the test dataset. This is an important plot that captures the relation between RUL and state of health (SoH) indicated by battery capacity. We observe that all models follow closely the trace of capacity degradation when the battery cycle number increases. However, TCN performs the best among all models. Fig. 4.b and Fig. 4.c show the true and predicted test values (RUL and cell temperature) for all three models. We observe that cell temperature prediction quality is not very good when the true temperature values vary a lot - particularly in the range 30,000 to 48,000 time steps. For better clarity, the prediction errors from Fig. 4.b and Fig. 4.c are shown in Fig. 5. We observe that the proposed TCN model has the smallest prediction errors.

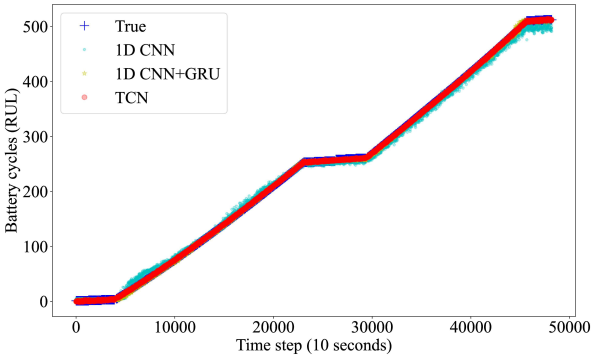
In the second phase of our experiments, all three models that were developed and tuned before, are again trained and tested on the NCA dataset. In this way, we wanted to verify how the models perform when they are retrained with a new dataset. The results of testing are shown in Table IV. We observe the most noticeable degradation for RUL prediction is for the 1D CNN model, while the performance of the other two models remained more or less the same. The 1D CNN+GRU model experienced the most observable degradation for cell temperature prediction. The TCN model has actually slightly improved in terms of prediction of cell temperature. The reason is that the cell temperature variation is smoother in the NCA dataset than in the NMC dataset.

VI. TinyML Models Evaluation

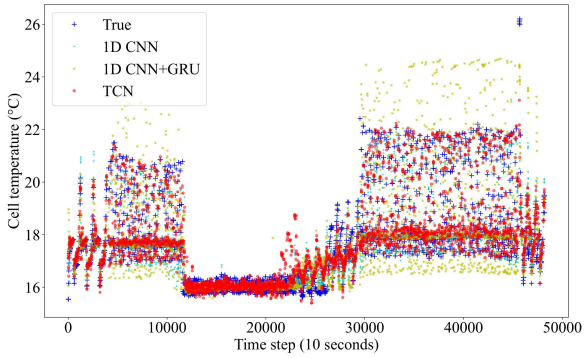
In this section, we further optimize the models and then deploy them as tinyML models on the IoT microcontroller device (Arduino Nano 33 BLE Sense board). We report results for the 1D CNN and TCN models only because the 1D CNN+GRU models could not be converted to TFLITE due to the GRU



(a)



(b)

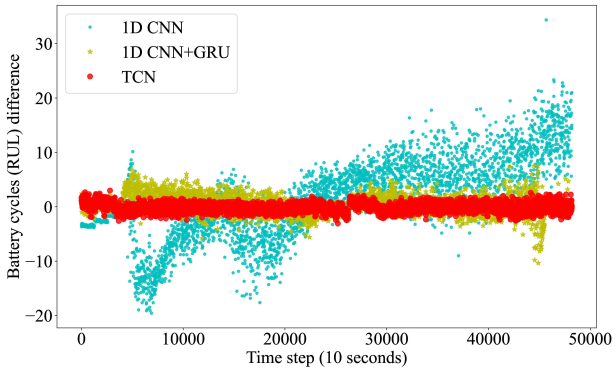


(c)

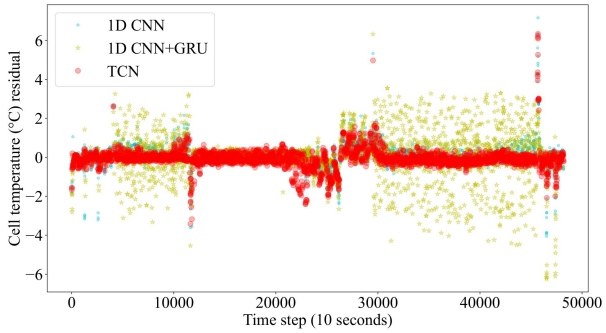
Fig. 4: Prediction results obtained with 1D CNN, 1D CNN+GRU, and TCN models (NMC dataset).

TABLE IV: RUL and cell temperature (CT) prediction - performance comparison using NCA dataset.

Models	Time steps	MAE	MSE	RMSE	R2
1D CNN (RUL)	6	9.57 ± 1.32	168 ± 60	12.79 ± 2.28	0.99 ± 0
1D CNN (CT)	18	0.35 ± 0.02	0.28 ± 0.02	0.53 ± 0.02	0.97 ± 0
1D CNN+GRU (RUL)	18	1.49 ± 0.12	4.23 ± 0.85	2.05 ± 0.21	1 ± 0
1D CNN+GRU (CT)	3	0.74 ± 0.73	4.1 ± 7.05	1.45 ± 1.41	0.75 ± 0.43
TCN (RUL)	12	0.89 ± 0.07	1.27 ± 0.2	1.12 ± 0.09	1 ± 0
TCN (CT)	18	0.12 ± 0.01	0.04 ± 0.01	0.19 ± 0.02	1 ± 0



(a)



(b)

Fig. 5: Prediction errors (as difference between predicted and true values).

TABLE V: Model size reduction via quantization.

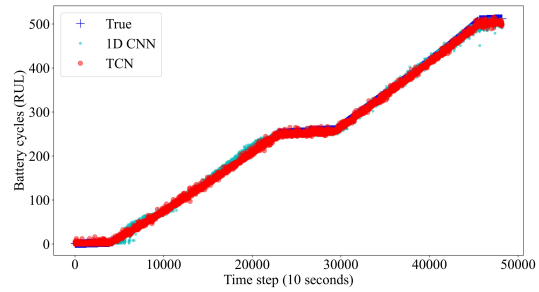
Models	TF model size (KB)	TFLITE model size (KB)	Reduction
1D CNN (RUL)	656	26.91	24x
1D CNN (Cell Temp.)	508	13.07	39x
TCN (RUL)	1208	40.59	30x
TCN (Cell Temp.)	1239	57.84	21x

units. The optimization consists of the conversion of the 32-bit float TensorFlow (TF) models from the previous section into 8-bit integer TF Lite (TFLITE) models. This optimization reduced the model sizes as reported in Table V. This reduction is what makes possible for the TFLITE models to be deployed on microcontrollers next. The TFLITE models are further converted to what is called TFLITE_MICRO models using the *xxd* tool. Finally the TFLITE_MICRO models are integrated in specific Arduino applications (developed in C/C++) and deployed on the Arduino Nano 33 BLE Sense board.

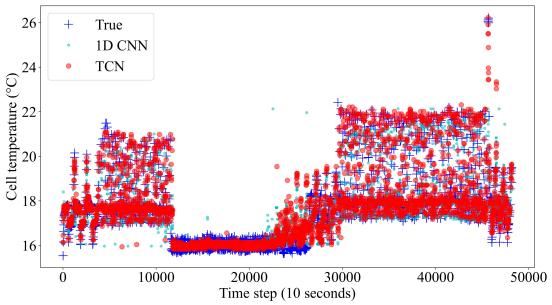
We compare the performance of TF, TFLITE, and TFLITE_MICRO models in Table VI. Note that the results for the TF and TFLITE models are obtained in Python using TensorFlow simulations. The results for the TFLITE_MICRO models are obtained with the models being executed on the hardware Arduino Nano 33 BLE Sense board. We observe that the performance degrades as we move from using TF to

TABLE VI: Performance comparison of TF, TFLITE, and TFLITE_MICRO models.

Models	TF MAE	TFLITE MAE	TFLITE_MICRO MAE
1D CNN (RUL)	5.91	6.16	6.173
1D CNN (Cell Temp.)	0.35	0.35	0.363
TCN (RUL)	0.64	1.88	4.213
TCN (Cell Temp.)	0.24	0.25	0.256



(a)



(b)

Fig. 6: Performance measured on the Arduino board for the TFLITE_MICRO models.

TFLITE to TFLITE_MICRO models. This is expected because the optimized TFLITE and TFLITE_MICRO models have all their inputs and weights represented on 8 bits. Fig. 6 shows the prediction result obtained with the TFLITE_MICRO models. We can see a noticeable difference in performance when compared to Fig. 4. This degradation in performance is reported in Table VI. We note that two TCN models for predicting RUL and cell temperature, whose performance is reported in Table VI, are similar (same structure and hyperparameters) except that the TCN model for predicting the cell temperature has the dilation feature in the convolutional layers removed. We removed the dilation because it resulted in better results. In terms of actual inference time, all TFLITE_MICRO models perform one inference in about 5 milliseconds measured on the Arduino Nano 33 BLE Sense board.

VII. Conclusion

We investigated 1D CNN, 1D CNN+GRU, and novel temporal convolutional network (TCN) models for prediction of the remaining useful life and of cell temperature of lithium-ion batteries. The proposed TCN models were optimized and deployed on IoT device microcontrollers using tinyML technologies. Extensive simulation and hardware experiments using Arduino Nano 33 BLE Sense board demonstrated that the proposed TCN models offer the best performance. In our future work, we plan to develop similar deep machine learning models to predict multiple variables of interest (such as RUL, state of health, state of charge, and temperature) to further reduce the memory capacity required to store and use for inference such models.

References

[1] N. Arandia, J.I. Garate, and J. Mabe, "Embedded sensor systems in medical devices: requisites and challenges ahead," *Sensors*, 2022.

[2] A.K.M.A. Habib, M.K. Hasan, G.F. Issa, D. Singh, S. Islam, and T.M. Ghazal, "Lithium-Ion battery management system for electric vehicles: constraints, challenges, and recommendations," *Batteries*, 2023.

[3] X. Meng, W. Wang, X. Cao, G. Li, and De. Li, "A RUL prediction method of lithium-ion battery based on extreme learning machine ," *J. Phys.: Conf. Ser*, 2023.

[4] Z. Xue, Y. Zhang, C. Cheng, and G. Ma, "Remaining useful life prediction of lithium-ion batteries with adaptive unscented Kalman filter and optimized support vector regression," *Neurocomputing*, 2020.

[5] Y. Cai, W. Li, T. Zahid, C. Zheng, Q. Zhang, and K. Xu, "Early prediction of remaining useful life for lithium-ion batteries based on CEEMDAN-transformer-DNN hybrid model," *Heliyon*, Vol. 9, 2023.

[6] L. REN, L. ZHAO, S. HONG, S. ZHAO, H. WANG, and L. ZHANG, "Remaining useful life prediction for lithium-Ion battery: a deep learning approach," *IEEE Access*, Vol. 6, 2018.

[7] M. Alipour, S.S. Tavallaey, A.M. Andersson, and D.Brandell, "Improved battery cycle life Prediction using a hybrid data-driven model incorporating linear support vector regression and Gaussian," *Chemphyschem*, 2020.

[8] X. Tang, H. Wan, W. Wang, M. Gu, L. Wang, and L. Gan, "Lithium-Ion battery remaining useful life prediction based on hybrid model," *Sustainability*, 2023.

[9] G. Crocioni, D. Pau, and G. Gruosso, "Li-ion batteries releasable capacity estimation with neural networks on intelligent IoT microcontrollers," *IEEE Mediterranean Electrotechnical Conf.*, 2020.

[10] Y. Mazzi, H.B. Sassi, A. Gaga, and F. Errahimi, "State of charge estimation of an electric vehicle's battery using tiny neural network embedded on small microcontroller units," *Inter. J. of Energy Research*, Vol. 46, 2022.

[11] A.A. Miaari and H.M. Ali, "Batteries temperature prediction and thermal management using machine learning: An overview," *J. of Energy Reports*, Vol. 10, 2023.

[12] R. Sanchez-Iborra and A. F. Skarmeta, "TinyML-enabled frugal smart objects: challenges and opportunities," *IEEE Circuits and Systems Magazine*, Vol. 20, 2020.

[13] Y. Preger, H.M. Barkholtz, A. Fresquez, D.L. Campbell, B.W. Juba, J. Romàn-Kustas, S.R. Ferreira, and B. Chalamala, "Degradation of Commercial lithium-Ion cells as a function of chemistry and cycling conditions," *J. of The Electrochemical Society*, 2020.

[14] G.D. Reis, C. Strange, M. Yadav, and S. Li, "Lithium-ion battery data and where to find it," *J. of Energy and AI*, 2021.

[15] Arduino website, "Arduino Nano 33 BLE Sense," available on: *https://store-usa.arduino.cc/products/arduino-nano-33-ble-sense*, 2024.

[16] R. David, J. Duke, A. Jain, V.J. Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, S. Regev, R. Rhodes, T Wang, and P. Warden, "TensorFlow-lite micro: embedded machine learning on tinyml systems," *MLSys Conf.*, 2021.

[17] A.V.D. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," *Arxiv*, 2016.

[18] Y. Liu, J. Li, G. Zhang, B. Hua, and N. Xiong, "State of charge estimation of lithium-ion batteries based on temporal convolutional network and transfer learning," *IEEE Access*, 2021.

[19] A.P. Wibawa, A.B.P. Utama, H. Elmunsyah, U. Pujianto, F. A. Dwiyanto, and L. Hernandez, "Time-series analysis with smoothed convolutional neural network," *J. of Big Data*, 2022.